Avaya™ MultiVantage™ Call Center Software

Call Vectoring and Expert Agent Selection (EAS) Guide

555-230-714
Issue 1.0
May 2002

Notice

Every effort was made to ensure that the information in this document was complete and accurate at the time of printing. However, information is subject to change.

Preventing Toll Fraud

“Toll fraud” is the unauthorized use of your telecommunications system by an unauthorized party (for example, a person who is not a corporate employee, agent, subcontractor, or working on your company's behalf). Be aware that there may be a risk of toll fraud associated with your system and that, if toll fraud occurs, it can result in substantial additional charges for your telecommunications services.

Avaya Fraud Intervention

If you *suspect that you are being victimized* by toll fraud and you need technical assistance or support, call Technical Service Center Toll Fraud Intervention Hotline at +1 800 643 2353 for the United States and Canada. For additional support telephone numbers, see the Avaya Web site:

http://www.avaya.com

Select **Support**, then select **Escalation Lists US and International**. This Web site includes telephone numbers for escalation within the United States. For escalation telephone numbers outside the United States, click on **Global Escalation List**.

Providing Telecommunications Security

Telecommunications security (of voice, data, and/or video communications) is the prevention of any type of intrusion to (that is, either unauthorized or malicious access to or use of) your company's telecommunications equipment by some party.

Your company's “telecommunications equipment” includes both this Avaya product and any other voice/data/video equipment that could be accessed via this Avaya product (that is, “networked equipment”).

An “outside party” is anyone who is not a corporate employee, agent, subcontractor, or working on your company's behalf. Whereas, a “malicious party” is anyone (including someone who may be otherwise authorized) who accesses your telecommunications equipment with either malicious or mischievous intent.

Such intrusions may be either to/through synchronous (time-multiplexed and/or circuit-based) or asynchronous (character-, message-, or packet-based) equipment or interfaces for reasons of:

- Utilization (of capabilities special to the accessed equipment)
- Theft (such as, of intellectual property, financial assets, or toll-facility access)
- Eavesdropping (privacy invasions to humans)
- Mischief (troubling, but apparently innocuous, tampering)
- Harm (such as harmful tampering, data loss or alteration, regardless of motive or intent)

Be aware that there may be a risk of unauthorized intrusions associated with your system and/or its networked equipment. Also realize that, if such an intrusion should occur, it could result in a variety of losses to your company (including but not limited to, human/data privacy, intellectual property, material assets, financial resources, labor costs, and/or legal costs).

Your Responsibility for Your Company’s Telecommunications Security

The final responsibility for securing both this system and its networked equipment rests with you - an Avaya customer's system administrator, your telecommunications peers, and your managers. Base the fulfillment of your responsibility on acquired knowledge and resources from a variety of sources including but not limited to:

- Installation documents
- System administration documents
- Security documents
- Hardware-/software-based security tools
- Shared information between you and your peers
- Telecommunications security experts

To prevent intrusions to your telecommunications equipment, you and your peers should carefully program and configure:

- your Avaya-provided telecommunications systems and their interfaces
- your Avaya-provided software applications, as well as their underlying hardware/software platforms and interfaces
- any other equipment networked to your Avaya products.

Trademarks

Alcatel is a trademark of Compagenie Financiere Alcatel, Paris, France.

AUDIX, BCMSVu, Callmaster, Conversant, DEFINITY, and GuestWorks are registered trademarks of Avaya, Inc.

GTX is a trademark of the Global TeleExchange.

MCI is a registered trademark of MCI WorldCom.

Avaya and MultiVantage are trademarks of Avaya, Inc.

Ordering Information

Call: Avaya Publications Center
Voice +1 800 457 1235
Fax +1 800 457 1764
International Voice +1 410 568 3680
International Fax+1 410 891 0207

Write: Globalware Solutions
200 Ward Hill Avenue
Haverhill, MA 01835 USA
Attention: Avaya Account Manager

Web: http://www.avayadocs.com

E-mail: totalware@gwsmail.com

Order: Document No. 555-230-714, Issue 1.0
May 2002

Avaya Support

Avaya provides a telephone number for you to use to report problems or to ask questions about your contact center. The support telephone number is 1-800-242-2121 in the United States. For additional support telephone numbers, see the Avaya Web site:

http://www.avaya.com

Select **Support**, then select **Escalation Lists US and International**. This Web site includes telephone numbers for escalation within the United States. For escalation telephone numbers outside the United States, click on **Global Escalation List**.

Acknowledgment

This document was written by the CRM Information Development group.

Avaya MultiVantage™
Call Center Software
Call Vectoring and Expert Agent Selection Guide

Contents

Chapter 13: Network Call Redirection

Chapter 14: Attendant Vectoring

Chapter 15: Holiday Vectoring

Chapter 16: Meet-me Conference

Chapter 17: Expert Agent Selection

Appendix C: Considerations for the vectoring features

Appendix D: Troubleshooting vectors

Appendix E: Advanced multi-site routing

Appendix F: Advanced information forwarding

Appendix G: Functional differences for DEFINITY G2 and DEFINITY ECS Call Vectoring and EAS

About this document

This guide discusses Call Vectoring and Expert Agent Selection (EAS) features of Avaya MultiVantage™ Call Center Software.

The following table gives a brief description of each chapter and appendix in this book.

Title/Page	Contents
Call Vectoring overview on page 19	High-level description of vectoring and EAS. Includes exercises and reading that is prerequisite to attending Instructor-led course(s).
Creating and editing call vectors on page 29	Brief tutorial and examples on how to create vectors using the SAT terminal interface.
Call Vectoring fundamentals on page 41	Additional depth of information regarding Call Vectoring and how the feature works.
Call Vectoring applications on page 61	Examples of how Call Vectoring can be implemented on-site.
Basic Call Vectoring on page 105	Detailed information on the Basic Call Vectoring option, including commands specific to the option.
Advanced Vector Routing - EWT and ASA on page 125	Detailed information on the Advanced Vectoring Routing option, including commands specific to the option.
ANI /II-digits routing and Caller Information Forwarding (CINFO) on page 139	Detailed information on vectoring use of ANI/Information Indicator-Digits and CINFO, including commands specific to these options.
Information Forwarding on page 151	Detailed information on the use of the Information Forwarding option, including commands specific to the option.
Adjunct (ASAI) Routing on page 163	Detailed information on Call Vectoring use of ASAI routing capabilities.
Call Prompting on page 181	Detailed information on the use and implementation of the Call Prompting option, including commands specific to the option.
Look-Ahead Interflow (LAI) on page 203	Detailed information on Call Vectoring use of LAI, including special considerations and troubleshooting.

Title/Page	Contents
Best Service Routing (BSR) on page 229	Detailed information on implementing BSR, including examples, troubleshooting, and BSR vector-writing tips.
Network Call Redirection on page 283	Detailed information on implementing NCR, including interactions with other contact center features.
Attendant Vectoring on page 299	Detailed information on the use of Attendant Vectoring option, including commands specific to the option. Note that Attendant Vectoring is used in non-contact center environments.
Holiday Vectoring on page 317	Detailed information on the use of the Holiday Vectoring option, including commands specific to the option.
Meet-me Conference on page 327	Detailed information about the commands used to create a Meet-me Conference vector, including an example Meet-me Conference VDN and vector.
Expert Agent Selection on page 341	Detailed information on the EAS feature, including interactions with other features and examples of implementation.
Appendix A: Call Vectoring commands on page 387	Complete, detailed list and definition of each vectoring command, including a Job Aid.
Appendix B: Vector management and monitoring on page 469	Additional information on implementing and upgrading to Call Vectoring.
Appendix C: Considerations for the vectoring features on page 479	Additional considerations for Basic Call Vectoring, Call Prompting, Adjunct Routing, and VDNs.
Appendix D: Troubleshooting vectors on page 493	Error messages.
Appendix E: Advanced multi-site routing on page 531	Detailed information on BSR and advanced routing to multiple sites.
Appendix F: Advanced information forwarding on page 545	Detailed information about ISDN (BRI or PRI) trunk group setting interactions with Information Forwarding, UCID, and Multi-Site Routing
Appendix G: Functional differences for DEFINITY G2 and DEFINITY ECS Call Vectoring and EAS on page 551	Differences between the DEFINITY G2 and DEFINITY ECS Call Vectoring.

Title/Page	Contents
Appendix H: Call Vectoring/EAS and BCMS/CMS interactions on page 563	Information regarding how CMS and BCMS report on vectoring.
Appendix I: Operation details for the route-to command on page 579	Detailed information on use of the **`route-to`** command.
Appendix J: Call flow and specifications for converse – VRI calls on page 585	Detailed information on vectoring and VRI calls.
Appendix K: Security issues on page 595	Issues to be aware of regarding the security of your site in relation to the use of Call Vectoring.
Appendix L: Setting up a call center on page 599	Worksheets to assist in the initial set up of a contact center.
Appendix M: Converting a Call Center to EAS on page 617	Worksheets to assist in the implementation of EAS in a contact center.
Appendix N: Feature Availability on page 625	Information about the MultiVantage options required to enable various vector commands.
Appendix O: Improving performance on page 627	Tips on improving the performance of vectors.

Intended audience and use of the guide

The guide is intended primarily for personnel who use Call Vectoring and/or EAS. You should use this guide as an information source for implementing Call Vectoring and/or EAS. A knowledge of Automatic Call Distribution (ACD) is assumed.

The level of your expertise in Call Vectoring and/or EAS should determine how you use the guide. Users who are unfamiliar with Call Vectoring should read Call Vectoring overview on page 19, then study Creating and editing call vectors on page 29. Users who will be using EAS should read Call Vectoring overview on page 19 and Expert Agent Selection on page 341. Finally, advanced users of Call Vectoring and/or EAS may only find it necessary to periodically reference a specific appendix (such as Call Vectoring commands on page 387, which contains a set of Call Vectoring/EAS command “manual pages”) to get the information needed.

Users who want to set up a call center (EAS and non-EAS) should read Appendix L: Setting up a call center on page 599, and users who want to convert a contact center to EAS should read Appendix M: Converting a Call Center to EAS on page 617.

Reasons for reissue

This document is being reissued for the following reasons:

- To be determined.
- Since system capacities change often, the capacity tables have been removed from this document. To determine the maximum values you can use in Call Vectoring commands, see *Avaya MultiVantage Definity Capacities Table*, 555-233-605. To access the document online, go to:

  http://avayadocs.com

Related documents

The following documents may include information related to the ACD feature.

Administration documents

The primary audience for these documents consists of switch administrators who work for external customers and for Avaya's dealers. The satisfaction and needs of our external customers is the primary focus for the documentation.

- *Administrator Guide for Avaya MultiVantage Software*, 555-233-506 – Provides complete step-by-step procedures for administering the switch, plus feature descriptions and reference information for SAT screens and commands.
- *Avaya MultiVantage Little Instruction Book for Basic Administration*, 555-233-756 – Provides step-by-step procedures for performing basic switch administration tasks. Includes managing phones, managing features, and routing outgoing calls.
- *Avaya MultiVantage Little Instruction Book for Advanced Administration*, 555-233-757 – Provides step-by-step procedures for adding trunks, adding hunt groups, writing vectors and recording announcements.
- *Avaya MultiVantage Little Instruction Book for Basic Diagnostics*, 555-233-758 – Provides step-by-step procedures for baselining your system, solving common problems, reading alarms and errors, using features to troubleshoot your system, and contacting Avaya.
- *Overview for Avaya MultiVantage Software*, 555-233-767 – Provides a brief description of the features available with DEFINITY ECS. This book does not provide a general overview of the switch nor of basic telephony.
- *Reports for Avaya MultiVantage Software*, 555-233-505 – Provides detailed descriptions of the measurement, status, security, and recent change history reports available in the system and is intended for administrators who validate traffic reports and evaluate system performance. Includes corrective actions for potential problems.
- *Avaya MultiVantage/Definity Hardware Solutions Guide*, 555-233-200 – Provides hardware descriptions, system parameters, lists of hardware required to use features, system configurations, and environmental requirements.

Call Center documents

These documents are issued for Avaya call center applications:

- *Avaya MultiVantage Call Center Software – Guide to ACD Call Center*, 555-230-716 – Provides feature descriptions and some implementation guidance for contact center features.
- *Avaya CMS Switch Connections and Administration*, 585-215-876 – Contains switch-to-CMS hardware connection diagrams and procedures to administer the switch-to-CMS link on the switch. It does not contain the administration of the CMS.
- *Avaya MultiVantage Basic Call Management System (BCMS) Operations*, 555-230-706 – Provides information on the use of the BCMS feature for ACD reporting.
- *Avaya MultiVantage Call Center Software – Call Vectoring Guide for BCS and Guestworks*, 555-230-715 – Provides information on how to write, use, and troubleshoot vectors on Category B products.

How to get help

For those times when you need additional help, the following help services are available. You may need to purchase an extended service agreement to use some of these help services. See your Avaya representative for more information.

- Avaya Centers of Excellence
  - Asia/Pacific
    +65-872-8686
  - Western Europe/Middle East/South Africa
    +441-252-391-889
  - Central/Eastern Europe
    +361-270-5160
  - Central/Latin America/Caribbean
    +1-303-538-4666
  - North America
    1-800-248-1111
- Avaya switch helpline
  1-800-225-7585
- Avaya Toll Fraud Intervention
  1-800-643-2353
- Avaya National Customer Care Center Support Line
  1-800-242-2121

Chapter 1: Call Vectoring overview

Call Vectoring overview provides the following information provides basic terminology and concepts associated with Call Vectoring and summarizes its benefits.

Call Vectoring overview includes the following topics:

What is Call Vectoring?

Call Vectoring is the process of defining vector programs that determine how a specific call should be routed and what call treatment that call is to be given.

Note:

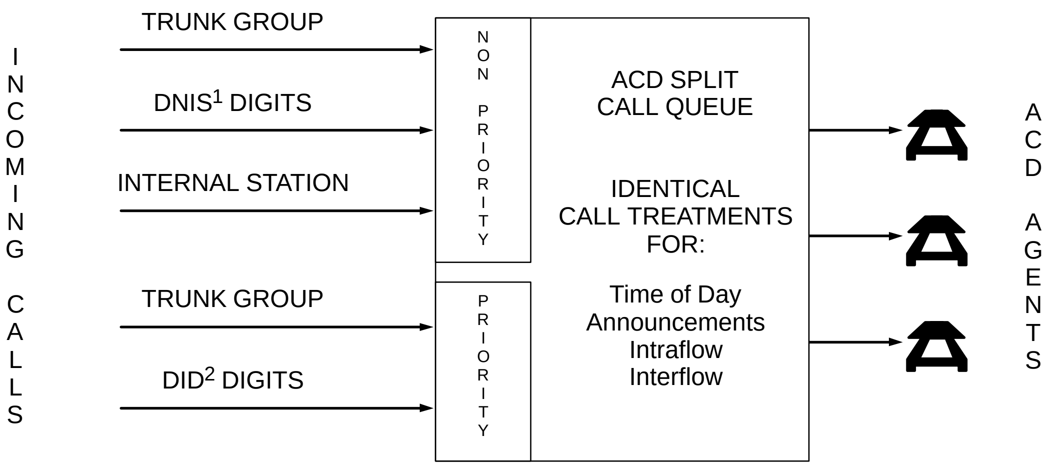
Sample vectors are provided throughout this manual to illustrate vectoring features and capabilities. Because they are simplified to clearly demonstrate specific features, they are not complete and should not be used without modification at your call center.

Call Vectoring provides a highly flexible approach for managing incoming call traffic to the switch. Using vectors, which are a series of user-defined commands, you can direct or route internal and network calls as desired in your call center and determine how these calls are processed. The processing of calls is known as call treatment. Calls can be directed to on-network or off-network destinations, to ACD agents, or to various other treatments. Call Vectoring also can be used with CallVisor ASAI.

Limitations of traditional ACD call processing

The traditional ACD approach is limited in the way it handles queued calls (that is, all calls within a specific queue receive identical announcements, intraflow parameters, and so forth). The following figure shows a simplified illustration of traditional ACD call processing.

Traditional ACD call processing



1. **Dialed Number Identification Service**
2. **Direct Inward Dialing**

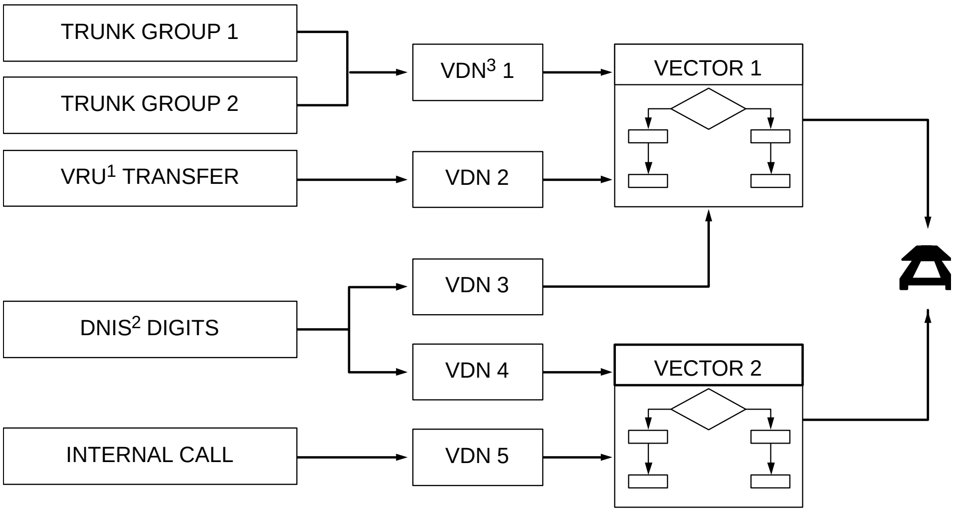
Call Vectoring, on the other hand, permits each call to be treated uniquely according to a number of factors, including the number the caller dials, the number the caller calls from, the number of calls in queue, and the time of day and/or day of the week. This even applies to all calls that are ultimately handled by the same agent group.

Call Vectoring is comprised of three basic components:

- Vector Directory Numbers
- Vectors
- Vector commands

Working together, these components direct incoming calls and ASAI event reports and requests to the desired answering destinations. They also specify how each call is processed. Call Vectoring may be set up as shown in the following figure.

Use of Call Vectoring for incoming calls



1. **Voice Response Unit**
2. **Dialed Number Identification Service**
3. **Vector Directory Number**

When a call arrives at a switch for which Call Vectoring is enabled, the call is first directed to a Vector Directory Number (VDN). A VDN is an internal telephone number that, in turn, directs the call to a specific vector. The VDN represents the call type or category, for example: billing, customer service, and so on. Thus, it defines the service that is desired by the caller. Multiple VDNs can point to the same or to different vectors, depending on whether the relevant calls are to receive the same or different treatment.

The vector is a set of commands that define the processing of a call. For example, a call can be queued and then routed to another destination.

The following screen shows an example of a vector.

```
1. goto step 3 if calls-queued in split 9 pri l < 20
      2. busy
      3. queue-to split 9 pri l
      4. wait-time 12 seconds hearing ringback
      5. announcement 2921
      6. wait-time 998 seconds hearing music
```

A vector can contain up to 32 command steps. Multiple vectors can be linked together to extend processing capabilities or to process calls to the same or different answering destinations. Any number of calls can use the same multiple vectors and process steps independently.

Understanding your goals and planning your system before you begin writing vectors is crucial. A planning guide is provided in Appendix L: Setting up a call center on page 599.

Call Vectoring options

Call Vectoring provides the following options:

- **Basic Call Vectoring** provides you with the ability to write vector steps that program the type of processing applied to a call by arranging a set of vector commands in the desired sequence. Depending on the command, you can do the following:
  - — Place the call in queue until an agent is available to answer the call.
  - — Provide a recorded information or delay announcement to the caller.
  - — Allow the caller to leave a recorded message.
  - — Access a Voice Response Unit (VRU) to start a script.

  See Basic Call Vectoring on page 105 for more information.
- **G3V4 Enhanced** provides for the following:
  - — Specification of a priority level with the oldest-call-wait conditional.
  - — Use of enhanced comparators (<>, >=, and <=).
  - — Use of wildcards in digit strings for matching collected digits and ANI or II-digits.
  - — Use of Vector Routing Tables.
  - — Multiple Audio/Music Sources for use with the **`wait-time`** command.
  - — Use of the **`interflow-qpos`** conditional with the **`goto`** and **`route-to`** commands to achieve FIFO or FIFO-like call processing
- **Advanced Vector Routing** allows you to route calls based on three additional conditions:
  - — Rolling Average Speed of Answer for a split, skill, or VDN.
  - — Expected Wait Time for a split/skill or for a call.
  - — The number of calls that are active in a specified VDN.

  See Advanced Vector Routing - EWT and ASA on page 125 for more information.
- **ANI/II Digits Routing** allows you to route calls based on either:
  - — The caller identity (ANI).
  - — The type of line where the call was originated (II-digits).

  See ANI /II-digits routing and Caller Information Forwarding (CINFO) on page 139 for more information.

- **Call Information Forwarding (CINFO)** allows you to collect caller-entered digits (ced) and customer-database-provided digits (cdpd) from the network. These digits can then be used in the same way as digits that are collected with Call Prompting.

  See ANI /II-digits routing and Caller Information Forwarding (CINFO) on page 139 for more information.

- **Enhanced Information Forwarding** allows you to forward the following information with each call. This information includes ANI, II-Digits, CINFO, ASAI-provided user information, Look-Ahead Interflow (LAI) information (such as VDN name), Universal Call ID (UCID), Best Service Routing data, Collected Digits, and elapsed in-VDN time.

  See Information Forwarding on page 151 for more information.

- **Call Prompting** allows you to collect digits and give some call control to the caller. Specifically, this feature allows callers with touchtone or rotary telephones to enter digits that are subsequently processed by the vector. Among other tasks, Call Prompting allows the caller to do the following:

  — Select one or more options from a menu to access recorded information or be routed to the correct split or agent.

  — Enter an extension to which a call can be routed.

  — Provide the call center with data, such as a credit card number that the center can use to process the call. This data also can be displayed on the telephone of the agent who answers the call.

  See the Call Prompting on page 181 for more information.

- **Look-Ahead Interflow** allows the call center to intelligently offload some or all calls to another ACD switch. When this feature is enabled, a call arriving at a vector that is unable to handle the call due to preset limits can interflow to a switch in a remote location whenever that switch is able to receive the call. By using this feature, you can establish a “load balancing” of calls among multiple locations primarily for lower-traffic, reduced-agent staffing periods. By using a certain conditional in a vectoring command, you can reduce processing and achieve First-In First-Out (FIFO) call distribution across the call center sites.

  See Look-Ahead Interflow (LAI) on page 203 for more information.

- **Best Service Routing (BSR)** allows the switch to compare specified splits or skills, determine which will provide the best service to a call, and deliver the call to that resource. If no agents are currently available in that split or skill, the call is queued. BSR is available in single-site and multi-site versions. Single-site BSR compares splits or skills on the switch where it resides to find the best resource to service a call. Multi-site BSR, which is activated via the Look-Ahead Interflow feature, extends this capability across a network of switches. Multi-site BSR compares local splits or skills, remote splits or skills, or both, and routing calls to the resource that provides the best service. BSR monitors the status of the specified resources and adjusts call processing appropriately to respond to changing conditions and operate more efficiently.

  See Best Service Routing (BSR) on page 229 for more information.

- **Network Call Redirection (NCR)** provides you with the ability to re-route calls on the public network.

  See Network Call Redirection on page 283 for more information.

- **Adjunct Routing** provides you with a means of evaluating calls before the calls are processed and implementing complex call center applications. Specifically, this feature allows a switch to request instructions from an associated adjunct, which is a processor that performs one or more tasks for another processor (the switch, in this case). The adjunct makes a routing decision according to agent availability and/or the caller information sent by the switch, and returns the routing response to the switch. By using this feature, the call center ensures that each call is delivered to the appropriate destination.

  See Adjunct (ASAI) Routing on page 163 for more information.

- **Attendant Vectoring** provides you with a means to route calls using call vectoring in an environment other than a traditional call center.

  See Attendant Vectoring on page 299 for more information.

- **Holiday Vectoring** provides you with a means to apply special call processing on days such as holidays or days when you are having special promotions.

  See Holiday Vectoring on page 317 for more information.

- **Meet-me Conference** provides you with a means to create dial-up, six-party conference calls using the features of Call Vectoring.

  See Meet-me Conference on page 327 for more information.

Benefits of Call Vectoring

Call Vectoring enables calls to be processed at a faster rate within an intelligent, real-time system, thereby providing appreciable cost saving to the user. The following table summarizes the benefits of Call Vectoring.

Call Vectoring benefits

Call Vectoring Benefits	Examples
Call Treatment	
Implement special treatment based on the time of day, the day of the week, and for holidays (for example, routing calls to a different vector when one location is on holiday).	Example application - customer service center on page 63 Conditional branching example on page 122 Example application - distributed call centers on page 70
Automatically change treatment according to either how long the call has been waiting or in response to changing traffic or staffing conditions.	Example application - automated attendant on page 64 Example application - mutual fund company on page 66 Example application - distributed call centers on page 70 Example application - help desk on page 71 Call interflow example on page 120 Using Call Prompting to route by collected digits on page 187 Using Call Prompting to branch by collected digits on page 188 Using LAI with route-to commands to outflow calls on page 210
Provide appropriate caller feedback during waiting (for example, music or announcements during heavy calling periods).	Delay announcement example on page 108 Supplementary delay announcement example on page 108 Forced announcement example on page 108 Information announcement example on page 109 Call delay with audible feedback on page 109 Call delay with multiple audio/music source feedback on page 110 Call delay with continuous audible feedback on page 110

Call Vectoring benefits (continued)

Call Vectoring Benefits	Examples
Provide multiple and/or recurring informational or delay announcements that are selected according to the time of day/day of the week, call volume, or staffing conditions.	Example application - customer service center on page 63 Leaving recorded messages (VDN as the coverage point option) on page 117 Call interflow example on page 120 Using LAI with route-to commands to outflow calls on page 210
Provide 24 hour/day, 7 day/week automated information announcements.	Information announcement example on page 109 Call delay with audible feedback on page 109
Remove selected calls (by providing busy or disconnect).	Busy command example on page 112 Call disconnect example on page 113 Accessing voice response scripts on page 114 Leaving recorded message on page 118 Unconditional branching example on page 122
Set up and test, in advance, special call treatments for events such as sales, advertising campaigns, holidays, snow days, and so on.	Information announcement example on page 109 Setting up a Holiday Table on page 321 Holiday Vectoring example 1 on page 323 Holiday Vectoring example 2 on page 324
Provide the caller with a menu of choices.	Example application - mutual fund company on page 66 Example application - help desk on page 71 Using Call Prompting to route by collected digits on page 187 Using Call Prompting to pass digits to an adjunct on page 193 Using dial-ahead digits to bypass announcements, example 1 on page 197
Queue calls to up to three splits simultaneously, consequently improving the average speed of answer and agent productivity.	Example application - customer service center on page 63 Example application - distributed call centers on page 70 Multiple split queuing example on page 116

Call Vectoring benefits (continued)

Call Vectoring Benefits	Examples
Implement routing to local or distant destinations.	Example application - customer service center on page 63 Example application - mutual fund company on page 66 Example application - distributed call centers on page 70 Example application - help desk on page 71 Call interflow example on page 120 Using Call Prompting to route by collected digits on page 187 Using Call Prompting to branch by collected digits on page 188 Using LAI with route-to commands to outflow calls on page 210
Connect callers to a voice-mail or messaging system either automatically or per caller request.	Example application - mutual fund company on page 66 Leaving recorded messages (VDN as the coverage point option) on page 117 Leaving recorded message on page 118
Call Routing	
Reduce call transfers by accurately routing callers to the desired destination.	Example application - mutual fund company on page 66 Using Call Prompting to route by collected digits on page 187 Using Call Prompting to branch by collected digits on page 188
Provide up to four ACD queuing priority levels and the ability to change the queuing priority dynamically, thereby, providing faster service for selected callers.	Example application - customer service center on page 63 Example application - mutual fund company on page 66 Example application - distributed call centers on page 70
Reduce agent and/or attendant staffing requirements by: (1) automating some tasks; (2) reducing caller hold time; (3) having agents in one split service multiple call types.	Example application - mutual fund company on page 66 Information announcement example on page 109 Call delay with audible feedback on page 109 Using Call Prompting to route by collected digits on page 187 Using dial-ahead digits to bypass announcements, example 1 on page 197 Using dial-ahead digits to bypass announcements, example 2 on page 198

Call Vectoring benefits (continued)

Call Vectoring Benefits	Examples
Information Collection	
Provide customized and/or personalized call treatment via information collection and messaging.	Example application - automated attendant on page 64 Example application - mutual fund company on page 66 Example application - help desk on page 71 Using Call Prompting to route by collected digits on page 187 Using Call Prompting to select options on page 191 Using dial-ahead digits to bypass announcements, example 1 on page 197
Collect information for use by an adjunct or by agent display.	Example application - help desk on page 71 Using Call Prompting to pass digits to an adjunct on page 193
Collect caller-entered or customer database-provided CINFO digits from the network.	CINFO example on page 149

Chapter 2: Creating and editing call vectors

This chapter gives you a practical start writing vectors. In this chapter you will learn the basic information that you need to write a representative vector and enter it online.

Methods for entering a vector online

A vector can be entered online using basic screen administration on the system administration terminal by any of the following three methods:

- Basic screen administration on the system administration terminal
- Avaya Call Management System (CMS)
- Avaya Visual Vectors

The following section discusses the basic screen administration method for entering a vector online at the switch system administration terminal. For instructions on creating a vector using the CMS interface, see *Avaya CMS Administration*, 585-215-515. For instructions on creating a vector with Visual Vectors, see *Avaya Visual Vectors User Guide*, 585-210-709.

Call Vector form — basic screen administration

A vector is entered online using basic screen administration by completing the Call Vector form. An example the first page of this form is shown in the following screen example.

Call Vector form (Page 1 of 3)

```
change vector 20                                              Page 1 of 3
                           CALL VECTOR
 Number: 20                Name:_______________________
Multimedia? n      Attendant Vectoring? n    Meet-me Conf? n          Lock? y
     Basic? y   EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? n   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? y      Holidays? y

 01 _______________
 02 _______________
 03 _______________
 04 _______________
 05 _______________
 06 _______________
 07 _______________
 08 _______________
 09 _______________
 10 _______________
 11 _______________
```

The following procedure summarizes how you can enter a vector online via basic screen administration.

1. Access the Call Vector Form by executing the **change vector *x*** command, where ***x*** is the number of the vector that you want to access. Use the change vector command either to change an existing vector or to create a new vector.

   If you are not certain of the number or name of a vector, enter the **list vector** command to view a complete list of all vectors that are administered for your system.

2. Assign a name to the vector by completing the blank next to the `Name` field. The vector name can contain up to 27 alphanumeric characters.

   Note:

   The vector number, which appears next to the `Number:` field, is automatically assigned by the system.

3. In the `Multimedia?` field, indicate whether the vector should receive early answer treatment for multimedia calls. Valid values are **y** or **n**.

   Note:

   This only applies if Multimedia Call Handling is enabled.

   - If you expect this vector to receive multimedia calls, set this field to **y.** The call is considered to be answered at the start of vector processing, and billing for the call starts at that time.
   - If you do not expect the vector to receive multimedia calls, set this field to **n**.

4. In the `Attendant Vectoring` field enter a **y** if the vector will be used as an attendant vector. Attendant Vectoring can be used only when enabled on the Customer Options form.

5. In the `Meet-me Conf` field enter a **y** if the vector will be used for the Meet-me Conference feature. Meet-me Conference can be used only when enabled on the Customer Options form.

   Note:

   Both Attendant Vectoring and Meet-me Conference cannot be enabled for a vector at the same time.

6. In the `Lock` field, indicate whether you will allow this vector to be displayed on and edited from a client application such as Visual Vectors.

   - If you enter **y**, the vector is locked and can only be displayed and modified in the switch administration software.
   - If you enter **n**, the vector is not communicated to client software such as Visual Vectors or CMS and may not be displayed and modified from these programs.
   - If Attendant Vectoring is enabled, the `Lock` field defaults to y and cannot be changed.

   Note:

   Always lock vectors that contain secure information, for example, access codes.

7. Look at the next fields and determine where a **y** (yes) appears. These fields indicate the Call Vectoring features and corresponding commands you can use. If an **n** (no) appears in one of these fields, you cannot use the corresponding feature.

   Note:

   The Call Vectoring features are optioned from the Customer Options form.

Basic	You can use the Basic Call Vectoring commands. See Basic Call Vectoring on page 105 for details on using these commands.
EAS	Expert Agent Selection is enabled. See Expert Agent Selection on page 341 for information on how the EAS feature works.
G3V4 Enhanced	You can use the G3V4 Enhanced Vector Routing commands and features. See Appendix N: Feature Availability on page 625 for an explanation of which features are included with G3V4 Enhanced Vector Routing.
ANI/II-Digits	You can use the ANI and II-Digits Vector Routing commands. See ANI /II-digits routing and Caller Information Forwarding (CINFO) on page 139 for details on using these commands. ANI/II-Digits Routing requires G3V4 Enhanced Vector Routing.
ASAI Routing	You can use the Adjunct Routing command. See Adjunct (ASAI) Routing on page 163 for details on using this command.
Prompting	You can use the Call Prompting commands. See Call Prompting on page 181 for details on using these commands.

LAI	Look-Ahead Interflow is enabled. See Look-Ahead Interflow (LAI) on page 203 information on how LAI works.
G3V4 Adv Route	You can use the G3V4 Advanced Vector Routing commands. See Advanced Vector Routing - EWT and ASA on page 125 for details on using these commands.
CINFO	You can collect ced and cdpd digits with the collect digits step. See ANI /II-digits routing and Caller Information Forwarding (CINFO) on page 139 for information on collecting these digits.
BSR	Best Service Routing (BSR) is enabled, and you can use the BSR commands. The available commands vary depending on whether you are using single-site or multi-site BSR. See Best Service Routing (BSR) on page 229 for information on the application of BSR.
Holidays	You can create tables to use for special days, such as holidays and promotional days. See Holiday Vectoring on page 317 for information on how to create holiday tables and define holiday vectors.

8. Enter a maximum of 32 vector commands in the blanks next to the step numbers. See Appendix A: Call Vectoring commands on page 387 for a complete description of all Call Vectoring commands.

   Note:

   You need not type every letter of each command that you enter. If you type just the first few letters of a command and press Enter or the Tab key, the system spells out the entire command.

9. Save the vector in the system by pressing Enter.

   Note:

   After editing a vector, verify that the vector will work as intended. This is particularly important if you deleted a step that was the target of a **`go-to`** step.

Inserting a vector step

To insert a vector step:

1. After entering the **`change vector`** command, press **F6** (Edit).
2. At the command line, type **`i`** followed by a space and the number of the step that you want to add and press Enter. For example, to insert a new vector step 3, type **`i 3`** and press Enter. You cannot add a range of vector steps.
3. Type the new vector step.

When a new vector step is inserted, the system automatically renumbers all succeeding steps and renumbers **`goto`** step references as necessary. Under certain conditions, attempts to renumber **`goto`** step references will result in an ambiguous renumbering situation. In this case, the step reference is replaced by an asterisk (*). You will receive a warning indicating that you must resolve the ambiguous references and your cursor automatically moves to the first reference that needs to be resolved. You cannot save a vector with unresolved goto references.

You cannot insert a new vector step if 32 steps are already entered in the vector. However, you can extend the vector program to another vector by using the **`goto vector unconditionally`** command at step 32.

Deleting a vector step

To delete a vector step:

1. After entering the change vector command, press **F6** (Edit)
2. At the command line, type **`d`** followed by a space and the number of the step you want to delete and press Enter. You can delete a range of vector steps. For example, to delete steps 2 through 5, type **`d 2-5`** and press Enter.

When a vector step is deleted, the system automatically renumbers all succeeding steps and renumbers **`go-to`** step references as necessary. Under certain conditions, attempts to renumber **`go-to`** step references will result in an ambiguous renumbering situation. In this case, the step reference is replaced by an asterisk (*).

For example, if a vector step that is the target of a goto step is deleted, the goto references are replaced by asterisks (*). For example, if you delete step 7 when you have a **`goto step 7 if`** vector step, the 7 is replaced by *.

You receive a warning indicating that you must resolve ambiguous references and your cursor automatically moves to the first reference that needs to be resolved. You cannot save a vector with unresolved goto references.

Creating and constructing a vector

Creating and constructing a vector provides a logical approach for vector construction. This method uses a starting vector that consists of one step and then builds on this vector to produce a new vector that provides additional functions. As each step is presented, you are introduced to one or more new vector commands or approaches to vector processing. While it is not practical to present all such commands and approaches, those presented in this tutorial should give you a good idea of how to use Call Vectoring.

Step 1: Queuing a call to the main split

If a call cannot be immediately answered by an agent or operator, the call is usually queued until an agent becomes available. A call can be connected to an available agent or queued via the vector shown in the following example. In this example, calls are queued to Split 5.

Queuing call to main split.

```
                                                           Page 1 of 1
                                 CALL VECTOR
 Number: 27           Name: base                Multimedia? n    Lock? n
Multimedia? n      Attendant Vectoring? n    Meet-me Conf? n          Lock? y
    Basic? y   EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? n   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? y      Holidays? y

 01 queue-to split 5 pri l
 02 _______________
 03 _______________
 04 _______________
 05 _______________
 06 _______________
 07 _______________
 08 _______________
 09 _______________
 10 _______________
 11 _______________
```

Agent Availability

If an agent is available, the **`queue-to split`** command automatically sends the call to the agent without queuing the call. However, if no agent is available, the command queues the call to the main split of agents. Once the call is sent to the main split queue, the call remains there until it is answered by an agent or some other treatment is provided.

Call Priority levels

Each call queued to a split occupies one queue slot in that split. Calls are queued sequentially as they arrive according to the assignment of the priority level. In our vector, note that the priority level low is assigned to the call. The priority level establishes the order of selection for each call that is queued. A call can be assigned one of four priority levels: top, high, medium, or low.

Within a given split (the main split, in our vector), calls are delivered to the agent sequentially as they arrive to the split queue and according to the priority level assigned. Accordingly, calls that are assigned a top priority (if any) are delivered to an agent first, calls that are assigned a high priority are delivered second, and so forth.

Step 2: Providing feedback and delay announcement

A call remains queued until an agent becomes available to answer the call. In the meantime, it is likely that the caller wants to hear some feedback assuring him or her that the call is being processed.

The vector shown in the following example provides one feedback solution. In this example, Announcement 2771 could contain this message: "We’re sorry. All of our operators are busy at the moment. Please hold.”

Providing feedback and delay announcement

```
                                                                  Page 1 of 3
                           CALL VECTOR
 Number: 27            Name: base                Multimedia? n    Lock? n
Multimedia? n      Attendant Vectoring? n    Meet-me Conf? n          Lock? y
    Basic? y   EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? n   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? y      Holidays? y

 01 queue-to split 5 pri l
 02 wait-time 10 seconds hearing ringback
 03 announcement 2771
 04 _______________
 05 _______________
 06 _______________
 07 _______________
 08 _______________
 09 _______________
 10 _______________
 11 _______________
```

Using the wait-time command

The **wait-time** command in step 2 provides a maximum 8-hour delay before the next vector step is processed. The time parameter can be assigned as follows:

- 0-999 secs
- 0-480 mins
- 0-8 hrs

In the example vector, the specified wait time is 10 seconds.

In addition to the delay period, the **wait-time** command provides the caller with feedback. In our vector, **ringback** is provided. Other types of feedback that can be provided with the **wait-time** command are: silence, system music, or an alternate music or other audio source. For more information see, wait-time command on page 109.

The **wait-time** command in the example vector provides the caller with a maximum of 10 seconds of ringback. If an agent answers the call before the **wait-time** command runs its course, the command is terminated, the delay period is ended and the accompanying feedback is stopped. In the current example, if the call is delivered to an agent after 4 seconds the caller does not hear the remaining 6 seconds of ringback.

If the call is not answered by the time the **wait-time** command is completed, vector processing continues.

The **announcement** command consists of a recorded message, and it is often used to encourage the caller to stay on the telephone or to provide information to the caller. If a call is delivered to an agent during the **announcement** command, the announcement is interrupted.

Multiple callers can be connected to an announcement at any time. See “Managing Announcements” in *Administrator Guide for Avaya MultiVantage Software*, 555-233-506, for more information about announcements.

Step 3: Repeating delay announcement and feedback

The announcement vector provides feedback to the caller after the call is queued. However, if the announcement is played and the agent does not answer the call soon after the announcement is complete, further feedback or treatment becomes necessary. One solution is provided in the following Call Vector example.

Repeating delay announcement and feedback

```
                                                           Page 1 of 1
                           CALL VECTOR
 Number: 27           Name: base                Multimedia? n    Lock? n
Multimedia? n      Attendant Vectoring? n    Meet-me Conf? n           Lock? y
     Basic? y   EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? n   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? y      Holidays? y

 01 queue-to split 5 pri l
 02 wait-time 10 seconds hearing ringback
 03 announcement 2771
 04 wait-time 60 seconds hearing music
 05 goto step 3 if unconditionally
 06 _______________
 07 _______________
 08 _______________
 09 _______________
 10 _______________
 11 _______________
```

The **`wait-time`** command in step 4 of this vector provides additional feedback (music) to the caller. If the call is not answered by the time step 4 is complete, the **`goto step`** command in step 5 is processed.

Conditional branching

Up to this point, we have discussed and illustrated Call Vectoring commands that cause sequential flow, that is, the passing of vector processing control from the current vector step to the next sequential vector step. The **`goto step`** command is an example of a Call Vectoring command that causes branching, that is, the passing of vector processing control from the current vector step to either a preceding or succeeding vector step.

The **`goto step`** command in vector step 5 allows you to establish an announcement-wait loop that continues until the agent answers the call. Specifically, the command makes an unconditional branch to the **`announcement`** command in step 3. If the call is not answered by the time that the announcement in step 3 is complete, control is passed to the **`wait-time`** command in step 4. If the call is still not answered by the time this command is complete, control is passed to step 5, where the unconditional branch is once again made to step 3. As a result of the established loop, the caller is provided with constant feedback.

Step 4: Queuing a call to a backup split

To this point, the vector example involves a call queued to one split. However, Call Vectoring allows a call to be queued to a maximum of three splits simultaneously, which improves can improve overall call response times. Multiple split queuing is especially useful during periods of heavy call traffic.

The vector shown in the following example allows a call to be queued to two splits.

Queuing call to backup split

```
                                                          Page 1 of 1
                                CALL VECTOR
  Number: 27           Name: base               Multimedia? n    Lock? n
 Multimedia? n      Attendant Vectoring? n    Meet-me Conf? n           Lock? y
      Basic? y   EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
  Prompting? n   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? y      Holidays? y

  01 queue-to split 5 pri l
  02 wait-time 10 seconds hearing ringback
  03 announcement 2771
  04 wait-time 10 seconds hearing music
  05 check split 7 pri m if calls-queued < 5
  06 wait-time 60 seconds hearing music
  07 announcement 2881
  08 goto step 5 if unconditionally
  09 _______________
  10 _______________
  11 _______________
```

The **queue-to split** command in step 1 queues the call to the main split. But if the call is not answered by the time the **wait-time** command in step 4 is complete, the **check split** command in step 5 attempts to queue the call to backup Split 7 at a medium priority. The condition expressed in the command (**if calls-queued < 5**) determines whether or not the call is to be queued to the backup split. Specifically, if the number of calls currently queued to Split 7 at a medium or higher priority is less than 5, the call is queued to the split.

Conditions used with the check split command

The **calls-queued** condition is one of several conditions that can be included in the **check split** command. The other conditions are **unconditionally**, **average speed of answer (rolling-asa)**, **available agents**, **staffed agents**, **expected wait time** and **oldest call waiting**. As is true for the **queue-to split** command, the **check split** command can queue a call at one of four priorities: **low**, **medium**, **high**, or **top**.

Elevating call priority

Note that if the call is queued to Split 7, the call priority is elevated from low to medium priority instead of a low priority, which is assigned if the call is queued by the **queue-to split** command in step 1. It is a good practice to raise the priority level in subsequent queuing steps to accommodate callers who have been holding the line for a period of time.

Step 5: Checking the queue capacity

A limited number of queue slots can be assigned to each split. Therefore, it is a good practice to check the main split queue for the number of calls that are already queued before allowing another call to queue to the split.

The number of slots assigned to each split is defined in the queue length field on the hunt group screen. A call that attempts to queue to a split with no available queue slots cannot be queued to that split and, accordingly, the **`queue-to split`** command fails. Vector processing would then continue with the next vector step. The following vector example includes provisions for checking queue capacity.

Checking queue capacity

```
                                                           Page 1 of 1
                              CALL VECTOR
 Number: 27            Name: base                Multimedia? n    Lock? n
Multimedia? n      Attendant Vectoring? n    Meet-me Conf? n          Lock? y
     Basic? y   EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? n   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? y      Holidays? y

 01 goto step 10 if calls-queued in split 5 pri l > 20
 02 queue-to split 5 pri l
 03 wait-time 10 seconds hearing ringback
 04 announcement 2771
 05 wait-time 10 seconds hearing music
 06 check split 7 pri m if calls-queued < 5
 07 wait-time 60 seconds hearing music
 08 announcement 2881
 09 goto step 6 if unconditionally
 10 busy
 11 _______________
```

A check of split 5 is implemented by the **`goto step`** command in step 1. In the example shown above, assume that 21 queue slots are assigned to split 5. Accordingly, the **`goto step`** command tests whether the split contains more than 20 calls via the condition **`if calls-queued in split 5 pri l > 20`**. If this test is successful, control is passed to the **`busy`** command, shown in vector step 10. The **`busy`** command gives the caller a busy signal and eventually causes the call to drop.

Alternately, if 20 or less medium priority calls are already queued to the main split when step 1 executes, the **`queue-to split`** command in step 2 queues the call, and vector processing continues at step 3.

Redirecting calls to a backup split

Instead of providing the caller with a busy tone if the **`queue-to split`** step cannot queue the call, the call can be queued to a backup split. To queue the call to another split, change the step parameter for the **`goto step`** command from 10 to 6 (so that the command reads **`goto step 6.....`**). In this case, control is passed from step 1 to the **`check split`** step (step 6). Because this queuing step is included within a continuous loop of steps (steps 6 through 9), continuous attempts to queue the call are now made.

Step 6: Checking for nonbusiness hours

If a caller calls during nonbusiness hours, you can still provide the caller with some information for calling back during working hours by playing the appropriate recorded message. This strategy is illustrated in the following Call Vector example. This vector would be used for a company that was open 7 days a week, from 8:00 a.m. to 5:00 p.m.

Checking for nonbusiness hours

```
                                                               Page 1 of 2
                                CALL VECTOR
 Number: 27            Name: base                Multimedia? n    Lock? n
Multimedia? n      Attendant Vectoring? n    Meet-me Conf? n          Lock? y
     Basic? y   EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? n   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? y      Holidays? y

 01 goto step 12 if time of day is all 17:00 to all 8:00
 02 goto step 11 if calls queued in split 5 pri l > 10
 03 queue-to split 5 pri l
 04 wait-time 10 seconds hearing ringback
 05 announcement 2771
 06 wait-time 10 seconds hearing music
 07 check split 7 pri m if calls-queued < 5
 08 wait-time 60 seconds hearing music
 09 announcement 2881
 10 goto step 6 if unconditionally
 11 busy
 12 disconnect after announcement 3222
```

The **`goto step`** command in step 1 checks if the call arrives during nonbusiness hours. Specifically, if the call arrives between 5:00 p.m. and 8:00 a.m. on any day of the week, the command passes control to step 12.

The **`disconnect`** command in step 12 includes and provides an announcement that first gives the caller the appropriate information and then advises him or her to call back at the appropriate time. The command then disconnects the caller.

If the call does not arrive during the specified nonbusiness hours, control is passed to step 2 and vector processing continues. On step 2, split 5 is checked for calls waiting at all priority levels.

Note:

As an alternative to disconnecting callers who place a call during nonbusiness hours, you can allow callers to leave a message by including the **`messaging split`** command within the vector. See Basic Call Vectoring on page 105 for more details.

Chapter 3: Call Vectoring fundamentals

The manner in which a call is processed depends how the switch is implemented and how the Call Vectoring software is implemented on the switch. The success of the call processing relies on:

- The resources that are available to process a call (for example: agents, splits, software, hardware). This is called call management.
- How the call is processed using vector processing, including VDN usage, vector control flow, and intelligent use of the vector programming capabilities.

This chapter describes these fundamental components of Call Vectoring.

Call management

When a call is placed to a switch enabled with Call Vectoring, the call is directed to an appropriate vector by means of a Vector Directory Number (VDN). A VDN is a “soft” extension number that is not assigned to an equipment location. A VDN maps to a single vector, but one or more VDNs can map to the same vector.

Once the call goes to a vector, call routing and treatment are determined by the commands in the vector. Processing starts at the first step and proceeds through the vector. Empty steps are passed over, and the vector process stops after the last step is reached.

However, one vector can direct the call to another vector or VDN, which in turn can direct the call to yet another vector, and so forth, up to a maximum of 1000 vector steps per call. When a call enters vector processing, a loop counter keeps track of the number of vector steps executed. If the loop counter exceeds 1000, a **`stop`** command is executed. However, when the **`interflow-qpos`** conditional is used, the execution limit is automatically increased to 3000 steps. This is because this conditional is designed to make rapid LAI loops practical.

The following sections discuss how calls are routed and queued by way of Call Vectoring. Subsequent sections discuss agent states, priority levels, caller feedback, and caller control.

Call flow

Calls enter a vector and execute steps sequentially beginning with step 1, unless there is a **`goto`** step. Most steps take microseconds to execute. The exception is steps with **`announcement`**, **`wait-time`**, and **`collect digits`** commands. A 0.2-second wait occurs after every seven executed steps unless an explicit wait has occurred. Note that **`wait-time`** with 0 seconds is not an explicit wait.

Call Vectoring uses several call flow methods to redirect and queue calls. These methods involve the use of the Call Vectoring commands, which are described later in this chapter. The methods for queuing and redirecting calls follow:

- **Multiple split queuing** allows a call to queue to up to three splits.
- **Intraflow** allows calls that are unanswered at a split within a predefined time to be redirected to one or more other splits on the same switch. If redirection depends on a condition to be tested, the process is referred to as conditional intraflow.
- **Interflow** allows calls that are directed to a vector to be redirected to an external or nonlocal split destination. This destination is represented by a number that is programmed in the relevant vector. Calls can be routed to an attendant or attendant queue, a local extension, a remote extension (Uniform Dialing Plan (UDP)), an external number, or a VDN.
- **Look-Ahead Interflow (LAI)** can be implemented for call centers with multiple ACD locations that are connected by way of ISDN PRI. This method allows a call to interflow only if a remote location is better equipped to handle the call. LAI can occur only when the proper conditions at the receiving switch are met.
- **Best Service Routing (BSR)** allows the switch to compare specified splits or skills, identify the split or skill that will provide the best service to a call, and deliver the call to that resource. If no agents are currently available in that split or skill, the call is queued. BSR is available in single-site and multi-site versions. Single-site BSR compares splits or skills on the switch where it resides to find the best resource to service a call. Multi-site BSR extends this capability across a network of switches, comparing local splits or skills, remote splits or skills, or both, and routing calls to the resource that will provide the best service.
- **Adjunct Routing** allows the switch to request a routing destination from an adjunct processor by way of ASAI. When this feature is enabled, the switch sends the ASAI adjunct a message that contains information about the calling party. The adjunct uses this information to determine, from its databases, the best place for the switch to send the call. The adjunct then passes this routing information back to the switch.

Caller control

Call Vectoring allows for the temporary transfer of call management control to the caller by several methods:

Caller-Selected Routing – This method prompts the caller to input information in the form of dialed digits from a touchtone telephone or from an internal rotary telephone that is located on the same switch. The capability is available if Call Prompting is enabled. A recorded announcement is usually used for prompting purposes. Once the caller inputs the digits, the call is routed to the correct department or destination. This procedure can significantly reduce the number of transferred calls and thus better satisfy the caller's needs.

In addition, if Call Prompting and Call Vectoring (CINFO) are enabled, the vector can collect caller-entered digits that are passed from the network by way of an ISDN message. These digits can be used to enhance caller control in the same way as digits that are collected directly by the switch.

Messaging – The caller can leave a voice message in the event that the call cannot be or has not yet been answered. When messaging is enabled, control is eventually passed to the AUDIX split.

Call queuing to splits

Basic Call Vectoring can queue calls to up to three splits simultaneously at any one of four priority levels. This process is called multiple split queuing. The first split to which a call is queued is called the main split, and the second and third split are designated as backup splits. Multiple split queuing enables more efficient utilization of agents, and thus provides better service to callers.

When an agent becomes available in any split to which the call is queued, the following events occur:

- The call is connected to the agent.
- The call is removed from any other queues. Announcements, music, ringback, or other audio source are terminated.
- Vector processing is terminated.

For more information about multiple split queuing, see Multiple split queuing on page 116.

Split queue priority levels

If Call Vectoring is not enabled, queued calls are tracked at one of two priority levels: Medium or High. If a call is queued using Call Vectoring, the call can be assigned one of four priority levels: Top, High, Medium, or Low. Within each priority level, calls are processed sequentially as they arrive.

Note:

A direct agent call is always given the highest priority, and is usually delivered before a call that is directed to a split. The exception is when skill-level Call Handling Preference is optioned and the skill that is administered to receive direct agent calls is not administered as the agent's highest skill level. A direct agent call is an ACD call that is directed to a specific ACD agent rather than to any available ACD agent in the split. For more information, see Direct agent calling on page 352.

Note:

If a call is already queued to one or more splits that are currently intended to serve as backup splits, the call could be requeued at the new priority level that is indicated in the command step. For more information on requeuing, see Call Vectoring commands on page 387.

Agent work mode

Call Vectoring can make call management decisions according to real-time agent work modes:

- Staffed-agents considers agents logged in to an ACD split.
- Available-agents considers agents logged in and ready to receive an ACD call.

These work mode states can appear as conditions within the **`check split`** and **`goto`** Call Vectoring commands, so that the commands can be made to check the number of staffed or available agents.

If a hunt group is not monitored, agents in the hunt group do not have log-in, log-out, or work modes. In such cases, staffed-agents is synonymous with administered, and available-agents is the number of agents who are ready to receive a hunt group call.

For ACD calls, agent states are further defined by the relevant work mode. The following list describes these modes:

- After-Call-Work Mode – The agent is unavailable to receive any ACD calls for any split. This mode can be used when the agent is doing ACD call-related work and can be implemented on a timed basis. This is known as Timed ACW. The system automatically places the agent into ACW after the agent completes a call that was received while in the manual-in work mode. In addition, the system can be administered through the Vector Directory Number or Hunt Group forms to automatically place agents into ACW for an administered period of time following the completion of each ACD call that is received while in the auto-in work mode.

- Auto-In Work Mode – The agent is available to receive calls and allows the agent to receive a new ACD call immediately after disconnecting from the previous call. When Multiple Call Handling is enabled, an agent in Auto-In Work Mode can elect to receive ACD calls by placing the active call on hold.
- Auxiliary-Work Mode – The agent is unavailable to receive any ACD calls for the specified split. This mode can be used when an agent is performing activities that are not associated with the ACD, such as going on a break.
- Manual-In Work Mode – The agent is available to receive calls. After the agent disconnects from an ACD call, they are automatically puts into the After Call Work Mode.

Note:

When Multiple Call Handling is enabled, an agent in Manual-In Work Mode can receive additional ACD calls by placing an active call on hold. For more information about agent work modes and Multiple Call Handling, see *Avaya MultiVantage Call Center Software – Guide to ACD Call Center*, 555-230-716.

Calling party feedback

The initial feedback a caller hears as the call is being processed by a vector depends on the origin classification of the call, which can be one of the following:

- Internal call from another switch user.
- Non-CO incoming call over a DID or tie trunk over which incoming digits are received.
- CO incoming call over a CO or automatic type tie trunk over which no digits are received.

For an internal or a non-CO call, the caller hears silence until one of the following vector steps is reached:

- For **`wait`** commands with system music, ringback, or an alternate music or audio source, the caller hears system music, ringing, or the music or audio associated with an administered port.
- For any **`announcement`** command, the caller hears the specified announcement. command is processed.
- For a **`busy`** command, the caller hears a busy signal.
- When the call rings a station, the caller hears ringback.

For a CO call, the caller hears CO ringback until one of the following vector steps is reached:

- Announcement (Caller hears the announcement.)
- Wait with system music or alternate audio/music source (Caller hears system music, or the music or audio associated with an administered port.)
- Call answered (Caller hears the agent or voice response answering the call.).

For a CO call that has answer supervision already supplied by way of the processing of an **announcement** or the issuing of a **wait-time** command, the caller may hear any of the following:

- Announcement when any **announcement** command is processed.
- Ringback, silence, system music, or an alternate audio or music source when a **wait-time** command is processed.
- Busy when a **busy** command is processed.
- Ringback when the call rings at a station.

Examples of how subsequent caller feedback is provided in a vector are provided in Basic Call Vectoring on page 105.

Dialed number identification service (DNIS)

In the traditional ACD arrangement, each agent in a given split is trained to answer calls that are relevant to one specific purpose. However, a contact center may wish to utilize agents trained to address multiple types of calls. This arrangement can allow resources to be used in a more efficient manner, with fewer agents overall and less administrative intervention by the ACD manager. For example, where 5 agents might be needed in each of three smaller splits (15 agents total) to handle 3 types of calls, only 11 or 12 agents might be needed in the combined split.

A network service known as Dialed Number Identification Service (DNIS) is available to exploit multi-skill agent capabilities. DNIS enables a unique multidigit number based on the dialed number associated with the call. The unique number may be sent to an agent, sent to a host computer with ASAI applications, used to provide different treatments for the call, and so forth.

The DNIS number is a function of the telephone number dialed by the caller. Each DNIS number in your telephone system can be programmed to route to an ACD split that is comprised of agents who are proficient in handling several types of calls.

Call Vectoring takes the DNIS number from the network and interprets this number as a VDN. When the call is delivered to the agent terminal, the unique name that is assigned to the particular VDN is displayed on the agent's terminal. This allows the agent to know the specific purpose of the call. As a result, the agent can answer with the appropriate greeting and be immediately prepared to service the customer.

Vector processing

If Call Vectoring is in effect, telephone calls are processed by one or more programmed sequences of commands called vectors.

Vector processing includes the following topics:

- Vector Directory Number (VDN)
- Vector control flow
- Programming capabilities.

Vector Directory Number

Within Call Vectoring, calls access the appropriate vector(s) by way of a Vector Directory Number (VDN). A VDN is a “soft” extension number that is not assigned to an equipment location. In effect, the digits dialed by a caller or sent to the switch from an external network are translated within the system as a VDN.

The VDN points to the vector, and it defines the service desired by the caller. The VDN also serves as the application number. It allows for specific call-handling and agent-handling statistical reporting within both the Basic Call Management System (BCMS) and the Avaya Call Management System (CMS) for each application that is handled by the call center.

VDNs are assigned to different vectors for different services or applications that require specific treatments. Any number of VDNs can point to the same vector. As a result, the same sequence of treatments can be given to calls that reach the system from different numbers or from different locations.

The VDN has several properties. These properties are administered on the Vector Directory Number form. The following screens show all possible fields available for a VDN. However, some fields display only when certain features are enabled.

```
change vdn xxxxx                                                   page 1 of 2
                      VECTOR DIRECTORY NUMBER

                            Extension: 2001
                                 Name: vdn 2001
                        Vector Number: 1
                  Attendant Vectoring? n
                   Meet-me Conference? n
                   Allow VDN Override? n
                                  COR: 1
                                   TN: 1
                             Measured: internal
       Acceptable Service Level (sec): 20
        VDN of Origin Annc. Extension:
                            1st Skill:
                            2nd Skill:
                            3rd Skill:
```

```
change vdn xxxxx                                                  page 2 of 2
                      VECTOR DIRECTORY NUMBER

                           Audix Name:
                 Messaging Server Name:
                    Return Destination:
                VDN Timed ACW Interval:
                       BSR Application:
         BSR Available Agent Strategy: 1st-found
                Conference Access Code:
                 Conference Controller:
'
```

The following list describes every field that is available on the VDN form and gives you information on the values that can be entered in the field.

- **Extension** — The extension number used to identify the VDN.
- **Name —** An alphanumeric name that identifies the VDN. This is an optional field that need not contain any data. The name may be truncated on agents' displays depending on the application. When information is forwarded with an interflowed call, only the first 15 characters are sent. The default is blank.
- **Vector Number** — An identification number that determines which vector is activated when a call comes into a VDN. Several VDNs may send calls to the same vector. The maximum Vector Number capacity varies by switch. For more information, see the Avaya switch capacity documents, which can be accessed online at:

  `http://www.avayadocs.com

- **Attendant Vectoring** — A y indicates that this is an Attendant Vectoring VDN. For more information, see Attendant Vectoring on page 299. This field defaults to y if Attendant Vectoring is the only Call Vectoring feature enabled on the customer options form. When removing a VDN, the switch verifies that this VDN is not being used on either the Console Parameters form or the Tenant Partitioning forms.
- **Meet-me Conference** — A **y** indicates that this is a Meet-me Conference VDN. For more information, see Meet-me Conference on page 327.
- **Allow VDN Override** — Valid entries are **y** and **n** (default). If a call is processed through multiple VDNs, this entry may affect the VDN name and various options assigned to the VDN. If it is set to n, the name of this VDN appears on the agent's display and the VDN's AUDIX mail is accessed. If any subsequent VDNs are used to process this call, their names do not appear on the terminating display and the AUDIX mail for the original VDN is accessed. If the field is set to y, the name of the VDN that appears on the terminating display depends on the administration and chaining of the subsequent VDNs and the AUDIX mail for the last VDN that is accessed. The default is n.
- **COR** — A 1-digit to 2-digit number that specifies the class of restriction (COR) to be assigned the VDN. The default value is 1. The field cannot be blank and must have an entry in the range from 0 to 95.
- **TN** — Enter the Tenant Partition number. The default value is 1.
- **Measured** — Used to collect measurement data for this VDN. Valid entries for Category A are **`internal`**, **`external`**, **both**, or **none**. External data collection is done by CMS, and internal data collection is done by BCMS. The default is none.

  Note:

  The BCMS feature must be enabled on the System-Parameters Customer-Options form for the Measured field to be set to `internal` or `both`. In addition, the appropriate CMS release must be administered on the Feature-Related System Parameters form if the field is being changed to `external` or `both`.

- **Acceptable Service Level (sec)** — Only displayed when the BCMS/VuStats Service Level option is enabled on the System-Parameters Customer-Options form and the Measured field is **internal** or **both**. Enter the number of seconds within which calls to this VDN should be answered. This will allow BCMS to print out a percentage of calls that were answered within the specified time. Valid entries are 0 to 9999 seconds. The default is blank.
- **VDN of Origin Annc. Extension** — This field is displayed only if VDN of Origin Announcements is enabled on the System-Parameters Customer-Options form. Enter the extension number of the VDN of Origin announcement. The default is blank.
- **1st/2nd/3rd Skill** — Only displayed when Expert Agent Selection is enabled on the System-Parameters Customer-Options form. Enter the desired Skill numbers in each field or leave the field blank. Valid entries are **1** to **999**. The default is blank.

- **AUDIX Name** — Only displayed for “r” model systems. If this VDN is associated with the AUDIX vector, enter the name of the AUDIX machine as it appears in the Adjunct Names form.
- **Messaging Server Name** — Not used in this release.
- **Return Destination** — The VDN extension number to which an incoming trunk call will be routed if it returns to vector processing after the agent drops the call. Valid entries are the VDN extension or blank. The default is blank.
- **VDN Timed ACW Interval** — When a value is entered in this field, an agent in auto-in work mode who receives a call from this VDN is automatically placed into After Call Work (ACW) when the call drops. Enter the number of seconds that the agent should remain in ACW following the call. When the administered time is over, the agent automatically becomes available. This field has priority over the Timed ACW Interval field on the Hunt Group form.
- **BSR Application** — This field is displayed only if Look-Ahead Interflow (LAI) and Vectoring (Best Service Routing) are enabled on the System Parameters Customer-Options form.To use multi-site Best Service Routing with this VDN, enter a 1- to 3-digit number to specify an application plan for the VDN.
- **BSR Available Agent Strategy** — This field is displayed only if Vectoring (Best Service Routing) is enabled on the System Parameters Customer-Options form. The available agent strategy determines how Best Service Routing identifies the best split or skill to service a call in an agent surplus situation. To use Best Service Routing with this VDN, enter an agent selection strategy in this field. Acceptable entries are **1st-found**, **UCD-LOA**, **UCD-MIA**, **EAD-LOA**, and **EAD-MIA**.
- **Conference Access Code** — A 6-digit Meet-me Conference access code or blank. Once a conference access code is assigned, an asterisk displays in this field for subsequent change display or remove operations by all users except the “init” superuser login. This field is displayed only when the VDN is a Meet-me Conference VDN.
- **Conference Controller** — A valid extension number or blank. If an extension number is entered, a user at that extension can change the access code for the Meet-me Conference VDN using a feature access code. If this field is blank, only a station user that is assigned with console permissions can change the access code for the Meet-me Conference VDN using a feature access code. In addition, remote access users can change a Meet-me Conference access code using the feature access code. This field is displayed only when the VDN is a Meet-me Conference VDN.

Display VDN for Route-To DAC? — The Display VDN for Route-to DAC option is designed to address situations where one of the following conditions is in effect:

- Either an **`route-to number`** or **`route-to digits`** vector command routes an EAS ACD agent call, with the coverage option set to **y**
- An adjunct routing step routes an EAS ACD agent call, with the coverage option set to **y**

The option facilitates the work of call center agents who respond to direct agent calls that originate from different Vector Directory Numbers (VDNs). When Display VDN for Route-to DAC is enabled, the name of the originating VDN is displayed at the agent station so that the agent can address the call in a more appropriate and efficient manner. For more information, see “Displaying VDN names for direct agent calls,” in Displaying VDN names for vector-initiated Direct Agent calls on page 480.

Implementation notes

The following list describes special situations due to the type of switch implementation that cause differences in the available fields on the VDN form.

- Data for the **Orig Annc** column appears only when VDN of Origin Announcement is enabled on the System-Parameters Customer-Options form.
- To list all VDNs using the same BSR Application Plan, enter the **list VDN BSR xxx** command (where xxx is the number of the BSR Application Plan used by one or more VDNs).

VDNs can be preassigned to incoming trunk groups, or they can be sent in digit form to the switch by a public or private network. The digits that are sent to the switch can come from the serving Central Office (CO) or toll office by way of the Direct Inward Dialing (DID) feature or DNIS. The digits can also come from another location by way of dial-repeating tie trunks, or they can be dialed by an internal caller. For a non-ISDN call, the last four digits of the number are sent to the system. For an ISDN call, the entire 10-digit number is sent to the system.

The last few digits of the destination passed to the switch/ACD on a DID or DNIS or on a dial tie-trunk call comprise the VDN. Automatic trunks do not pass destination address digits. Instead, each such trunk always routes to a specific incoming destination that is programmed for the corresponding automatic trunk group. The destination can be an attendant queue, an extension, a hunt group number, or a VDN.

VDN Override

VDN Override allows information about a VDN to which a call is routed to be used instead of the information about the current VDN. This information includes:

- The name of the subsequent VDN
- Skill sets
- Messaging split command with the “active” entry.
- VDN of Origin Announcement
- Tenant number
- VDN Timed ACW Interval
- VDN Return Destination with the condition that once the call leaves vector processing for the first time, the Return Destination never changes. For more information, see Appendix C: Considerations for the vectoring features on page 479.
- BSR Application
- BSR Available Agent Strategy

Note:

Throughout this document the “active” VDN is the active called VDN as modified by VDN override rules. The “latest” VDN is the most recent VDN to which the call was routed.

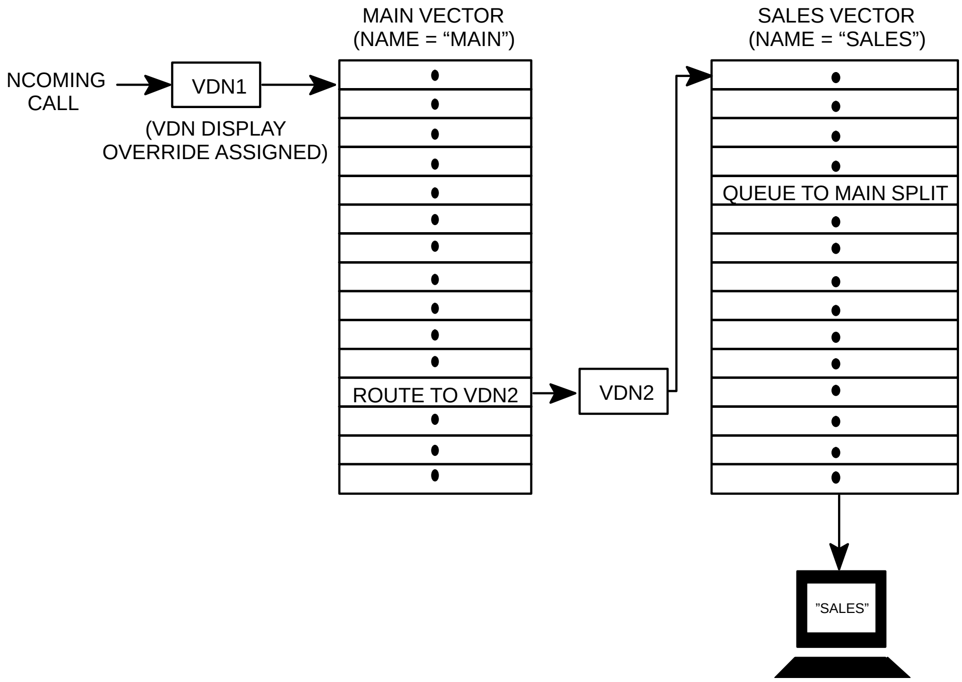
VDN Override can be used in conjunction with a vector that prompts the caller for a particular service. For example, a call is placed to an automobile dealer. Like most such dealers, this one consists of several departments, including “Sales” and “Parts.” Assume that the caller wants to talk to someone in “Sales.” In this case, the call comes into the “Main” vector (whose VDN name is “Main”) and is eventually routed to the “Sales” vector (whose VDN name is “Sales”). If VDN Override is assigned to the “Main” VDN, the “Sales” VDN name appears on the agent’s telephone display when the call is finally connected to the agent.

The VDN Override process is shown in Figure 3-1: VDN override assigned to originally called VDN on page 53. In this example, the “Sales” VDN is the active VDN as well as the latest VDN. If VDN override was not assigned to the “Main” VDN, the agent’s telephone display would show “Main.” In this case, “Main” is the active VDN while “Sales” is the latest VDN.

For Expert Agent Selection (EAS), if this field is y on the original VDN, the Skills of the new VDN are used. If this field is n on the original VDN, the Skills of the original VDN are used.

For Best Service Routing (BSR), if this field is y on the original VDN, the BSR Application and Available Agent Strategy of the new VDN are used. If this field is n on the original VDN, the BSR Application and Available Agent Strategy of the original VDN are used.

Figure 3-1: VDN override assigned to originally called VDN



VDN in a coverage path

A VDN can be assigned as the last point in a coverage path. Whenever a VDN is assigned as such, a call goes to coverage and can then be processed by Call Vectoring or Call Prompting if either is enabled. Accordingly, the Call Coverage treatment for the call is extended. Coverage can be sent to an external location or the type of coverage can be controlled by the caller.

VDN in a coverage path is used for a number of applications, including:

- Sending Direct Agent calls or personal calls to an agent in the EAS environment.
- Routing coverage calls off-premises using the **`route-to`** command.
- Serving as a coverage point for specific call operations. For example, sending calls to a secretary during the day and to AUDIX at night.

For more information, see Option with the VDN as the coverage point on page 117. For information about interactions, see *Administrator Guide for Avaya MultiVantage Software*, 555-233-506.

Redirect on No Answer to a VDN

The Redirection on No Answer (RONA) feature redirects a ringing ACD call after an administered number of rings. It prevents a call from ringing indefinitely at a terminal when an agent does not answer. When a call is redirected, the system puts the agent into AUX work so that the agent is no longer available to receive ACD calls. In the case of Auto-Available Splits, the system logs the agent out when a call is redirected.

A VDN can be administered as the destination of a RONA processed call. A call that is not answered can be redirected to a VDN to receive special treatment. Enter the number of the destination VDN for a RONA call in the Redirect to VDN field on the Hunt Group form. All calls that are redirected by RONA from that split are sent to the same administered VDN.

If no destination VDN is administered, but the number of rings for redirection is entered, the call redirects back to the split/skill.

Direct Agent calls that are not answered follow the agent's coverage path. If no coverage path is administered, calls are redirected to the VDN that is administered as the agent's first primary skill.

For more information, see the Redirection on No Answer section in *Avaya MultiVantage Call Center Software – Guide to ACD Call Center*, 555-230-716.

Service Observing VDNs

The Service Observing feature provides the option of being able to observe VDNs. With this option an observer selects a specific VDN and bridges onto calls (one call at a time) that have just started vector processing for that VDN. The observer hears all tones, announcements, music, and speech that the caller and the agent hear and say, including Call Prompting and caller dialing. Also, the observer hears VDN of Origin Announcements. Once the system makes an observing connection to a call in vector processing, it maintains the connection throughout the life of the call until the call is disconnected or until the observer hangs up. This is true even if the call is routed or transferred externally.

For more information about Service Observing VDNs, see the Service Observing section in *Avaya MultiVantage Call Center Software – Guide to ACD Call Center*, 555-230-716.

Vector control flow

The vector process starts at the first step in the vector and then proceeds sequentially through the vector unless a **`goto`** command is encountered. Any steps that are left blank are skipped, and the process automatically stops after the last step in the vector.

The Call Vectoring “programming language” provides three types of “control flow that pass vector-processing control from one vector step to another.The types of control flow are described in the following list:

- **Sequential flow** passes vector-processing control from the current vector step to the following step. Most vector commands allow for a sequential flow through the vector.

  Note:

  Any vector command that fails automatically passes control to the following step.

- **Unconditional branching** unconditionally passes control from the current vector step to either a preceding or succeeding vector step or to another vector. For example, **`goto step 6 if unconditionally`**.

- **Conditional branching** conditionally passes control from the current vector step to either a preceding and/or succeeding vector step or to a different vector. This type of branching is based on the testing of threshold conditions. For example, **`goto vector 29 if staffed-agents in split 6 < 1`**.

  Note:

  Call Vectoring has an execution limit of 1000 steps. Once a call enters vector processing, a “loop counter” keeps track of the number of vector steps executed. If the loop counter exceeds 1000, a **`stop`** command is executed. However, when the **`interflow-qpos`** conditional is used, the execution limit is automatically increased to 3000 steps. This is because this conditional is designed to make rapid LAI loops practical.

  Note:

  An implicit wait of 0.2 seconds is provided after every seven vector steps if vector processing is not suspended during any one of these steps. For more information, see Wait-time command on page 464.

Termination versus stopping

When vector processing is terminated, the call leaves the vector. Vector termination can result from a number of events, such as when a call is:

- Ringing at an agent's station
- Abandoned by the calling party
- Subject to a forced disconnect or a forced busy
- Successfully routed to an extension or to an off-premises number

The termination of vector processing termination differs from stopping, which is caused by the **`stop`** command or by the execution of the final step in the vector. Termination differs from stopping in the following ways:

- If a call is queued, termination removes the call from the queue.
- A **`stop`** command prevents the processing of new vector steps but leaves the call in queue and the calling party continues to receive feedback, such as ringback.
- If vector processing stops and the call is not queued, the call is dropped.

Programming capabilities

Call Vectoring commands perform various call-related functions, which include:

Providing call treatments

Audible feedback, including silence, ringback, system music, or an alternate audio or music source, or a busy tone can be provided to the caller. The caller can be provided with a recorded announcement to indicate that an agent is unavailable to answer the call or to provide other information or instructions. An Audix session can also be initiated.

Vector processing can be delayed for a specific number of seconds before the next vector step is executed. The call can also be disconnected, if necessary.

- **Routing calls**. Calls that are not immediately answered by an agent can be queued to one or more splits. A caller can also leave a recorded message if he or she chooses to do so. Finally, a call can be routed to a number programmed in the vector or to digits that are collected from the caller.
- **Branching/programming**. Branches can be made from one vector step to another such step or to another vector. This can be done unconditionally as well as conditionally. Conditional branching is done according to a number of conditions, for example, number of available agents in a split, number of calls in a split queue, the number of the phone the call is made from, and so forth. Finally, vector processing can be stopped when necessary.
- **Collecting and acting on information**. Optionally, touchtone digits can be collected and serve as the basis for further vector processing. For example, the caller can enter certain touchtone digits to reach a specific agent.
- **Executing VRU scripts**. Voice scripts on a VRU can be executed for the caller. Voice scripts provide the caller with information or instructions. The caller can often make an appropriate response to a voice script, for example, by entering touchtone digits.

Command summary

This section lists and describes the commands used by the Call Vectoring features. The list is meant to help familiarize the reader with these commands. The commands are also described in further detail in Call Vectoring commands on page 387.

- **Adjunct Routing** is available only when the CallVisor ASAI capabilities and Basic Call Vectoring are enabled on the switch. The command causes a message to be sent to an ASAI adjunct requesting routing instructions.
- **Announcement** provides the caller with a recorded announcement.
- **Busy** gives the caller a busy signal and causes termination of vector processing.

- **Check** conditionally checks the status of a split or skill for possible termination of the call to that resource. The command either connects to an agent in the split/skill or puts the call into its queue at the specified queuing priority level if the condition specified as part of the command is met. A call can be queued to up to three different splits/skills simultaneously.
- **Collect Digits** collects up to 16 digits that are either entered by the caller during vector processing, sent by the network, or received from an adjunct. An optional announcement can be played first when the digits are being collected directly from the caller.
- **Consider Location** obtains the Expected Wait Time (EWT) and agent data needed to identify the best remote location in multi-site Best Service Routing applications. One **`consider`** step must be written for each location that you want to check.
- **Consider Split/Skill** obtains the EWT and agent data needed to identify the best local split or skill in single-site Best Service Routing vectors. One **`consider`** step must be written for each split/skill that you want to check.
- **Converse-on Split** integrates Voice Response Units (VRUs) with the switch. Specifically, the command allows voice response scripts to be executed while the call remains in queue, and it allows the passing of data between the switch and the VRU.
- **Disconnect** ends treatment of a call and removes the call from the switch. The command also allows the optional assignment of an announcement that will play immediately before the disconnect.
- **Goto Step** is a branching step that allows conditional or unconditional movement to a preceding or succeeding step in the vector. Conditional branching is determined by a number of factors. For example: the number of calls that are queued in the split, the number of staffed agents who are in the split, if the call arrives at a time of day that is in a holiday table, and so on.
- **Goto Vector** is a branching step that allows conditional or unconditional movement to another vector. Conditional branching is determined by a number of factors. For example: the number of calls that are queued in the split, the number of staffed agents who are in the split, if the call arrives at a time of day that is in a holiday table, and so on.
- **Messaging Split** allows the caller to leave a message for a specified extension or the VDN extension.
- **Queue-to** unconditionally queues a call to a split or skill and assigns a queuing priority level to the call in case no agents are available. A call that is sent with this command either connects to an agent in the split or skill or enters its queue.
- **Queue-to attd-group** queues a call to a specified attendant group and is available only for attendant vectors. A call that is sent with this command either connects to an available agent within the group or enters the queue if no agent is available.
- **Queue-to attendant** queues a call to a specific attendant and is available only for attendant vectors. The call only queues to the agent if the agent is a member of the TN associated with the call.

- **Queue-to-hunt group** queues a call to up to three hunt groups. A call that is sent with this command connects to an agent in the hunt group or enters the hunt group queue.
- **Reply-best** returns data to another switch in response to a status poll. `Reply-best` is only used in status poll vectors in multi-site Best Service Routing applications.
- **Route-to Digits** routes the call to the destination that is specified by a set of digits that are collected from the caller or VRU by the previous `collect digits` step. For more information, see Appendix I: Operation details for the route-to command on page 579.
- **Route-to Number** routes the call to the destination specified by the administered digit string. Form more information, see Appendix I: Operation details for the route-to command on page 579.
- **Stop** terminates the processing of any subsequent vector steps.
- **Wait-Time** is used to specify whether the caller hears ringback, system music, silence, or an alternate audio or music source while the call is waiting in queue. The command also delays the processing of the next vector step by the specified delay time that is included in the command's syntax.

Condition testing within the commands

As was mentioned in the previous section, a number of the Call Vectoring commands are implemented according to a tested condition that comprises part of the command. In other words. If the condition that is expressed in the command is true, then the command action is executed. If the condition that is expressed in the command is false, then the command action is not implemented, and the next vector step is processed.

For more information about the syntax of each condition, see Appendix A: Call Vectoring commands on page 387.

The following list provides a set of conditions that might comprise the conditional portion of a Call Vectoring command:

Note:

The available set of conditions is dependent upon the optional features that are enabled. For more information, see Appendix N: Feature Availability on page 625.

- The number of staffed agents in a split
- The number of available agents in a split
- The number of calls queued at a given priority to a split
- The amount of time that the oldest call has been waiting in a split
- Whether or not a call receives special holiday processing
- The Average Speed of Answer for a split or a VDN
- The Expected Wait Time for a split or for a call that has entered vector processing
- A reduction in Expected Wait Time if a call is queued to a backup resource

- The number of calls in a queue that are eligible for interflow processing using **`interflow q-pos.`**
- The number of active calls that have been routed by a VDN
- The caller identity (ANI)
- The type of originating line (II-digits)
- The digits entered by the caller, sent in an ISDN message from the network (CINFO), or received from an ASAI or VRU adjunct
- The time-of-day and day of the week that the call is placed. The syntax for this condition can be illustrated as follows: **`mon 8:01 to fri 17:00`** means anytime between 8:01 a.m. Monday through 5:00 p.m. Friday, and **`all 17:00 to all 8:00`** means between 5:00 p.m. and 8:00 a.m. on any day of the week.

Depending on the condition, specific comparison operators and a threshold might be in effect. Examples of comparison operators are < (less than), > (greater than), = (equal to), <= (less than or equal to), >= (greater than or equal to), <> (not equal to), and “in” or “not-in”. A threshold is a range of accepted numerical entries.

The chapters on the Call Vectoring features illustrate condition checking in more detail.

Chapter 4: Call Vectoring applications

This chapter provides example applications of the Call Vectoring feature.

List of example applications

Example applications and the primary feature that is illustrated are listed in the following table.

Example	Features used
Customer service center example on page 63	Basic Call Vectoring
Automated attendant example on page 64	Call Prompting
Data in/voice answer and data/message collection example on page 65	Call Prompting, Basic Call Vectoring
Distributed contact centers example on page 69	Look-Ahead Interflow, Basic Call Vectoring
Help desk example on page 71	Adjunct Routing, Call Prompting, Basic Call Vectoring
Insurance agency/service agency example on page 72	Basic Call Vectoring, Call Prompting, Rolling ASA, EWT, VDN Calls, and ANI Routing
Warranty service (with EAS) example on page 76	Basic Call Vectoring, EAS
Resort reservation service (with EAS) example on page 79	Basic Call Vectoring, Adjunct Routing, Call Prompting, EAS
Local attendant group access code on page 84	Attendant Vectoring
Incoming trunk calls to attendant group on page 84	Attendant Vectoring
Incoming LDN calls on page 85	Attendant Vectoring
QSIG CAS example on page 85	Attendant Vectoring

Example	Features used
Night station service example on page 87	Attendant Vectoring
Holiday Vectoring example on page 88	Holiday Vectoring
Network Call Redirection example on page 90	BSR multi-site, NCR
BSR using EWT and agent adjustments example on page 92	BSR multi-site
Dial by Name on page 95	Basic Call Vectoring, Call Prompting

Customer service center example

The example scenario involves a customer service center that is open weekdays from 8 a.m. until 5 p.m. The center provides two separate telephone numbers. One number is for regular customers, while the other number is for priority customers. The following vector examples show how calls to the customer service center are handled.

Example application - customer service center

```
VDN (extension=1021  name=‘‘Customer Serv’’  vector=21)
Vector 21:
      1. goto vector 29 if time-of-day is all 17:00 to all 08:00
      2. goto vector 29 if time-of-day is fri 17:00 to mon 08:00
      3. goto step 10 if calls-queued in split 1 pri l > 10
      4. queue-to split 1 pri m
      5. wait-time 10 seconds hearing ringback
      6. announcement 3521
      7. wait-time 50 seconds hearing music
      8. announcement 3522
      9. goto step 7 if unconditionally
     10. busy

VDN (extension=1022  name=‘‘Priority Cust’’  vector=22)
Vector 22:
      1. goto vector 29 if time-of-day is all 17:00 to all 08:00
      2. goto vector 29 if time-of-day is fri 17:00 to mon 08:00
      3. goto step 12 if calls-queued in split 1 pri h > 10
      4. queue-to split 1 pri h
      5. announcement 3521
      6. wait-time 10 seconds hearing music
      7. check split 2 pri h if oldest-call-wait < 20
      8. check split 3 pri h if oldest-call-wait < 20
      9. announcement 3522
     10. wait-time 60 seconds hearing music
     11. goto step 7 if unconditionally
     12. route-to number 0 with cov n if unconditionally

No VDN
Vector 29:
      1. announcement extension 3529
      2. wait-time 10 seconds hearing silence
      3. disconnect after announcement 3529
```

When a priority customer places a call to the correct number, vector 22 is accessed. The first two steps of this vector determine if the call arrives during nonbusiness hours. If the call arrives between 5:00 p.m. and 8:00 a.m. on any given day, step 1 routes the call to Vector 29. Step 2 does the same if the call arrives during the weekend, that is, between 5:00 p.m. Friday and 8:00 a.m. Monday. If vector 29 is accessed, the caller is given the appropriate announcement twice (skills 1 and 3) and is then disconnected (step 3).

If the call is placed during business hours, step 3 of vector 22 determines if the number of high-priority calls that are queued in the main split exceeds 10. If more than 10 calls are in the queue, control is sent to step 12, which routes the call to the attendant. If less than 10 calls are in the due, the call is queued to the main split (step 4). If the call is not answered immediately, an appropriate announcement is provided (step 5), followed by a wait period (step 6).

If the call is not answered after the wait time specified in step 6, steps 7 and 8 attempt to queue the call to a backup split (splits 2 and 3, respectively). The call is queued to either split if the oldest call in the split has been waiting fewer than 20 seconds.

Even if the call is queued to one of the backup spilts, the call is passed to steps 9 through 11, which implement an announcement-wait cycle that continues until either an agent answers the call, or the caller abandons the call.

A call that is placed by a nonpriority customer is processed by vector 21. Vector 21 provides a treatment similar to that provided by vector 22, with the following exceptions:

- Backup splits are not queried for nonpriority calls
- Priority calls are assigned a higher priority in the queue
- Priority calls route to an operator when too many calls are queued, but nonpriority calls route to a busy signal.

Automated attendant example

This example scenario shows the use of Automated Attendant, which is one of the applications that can be supported by the Call Prompting feature. Automated Attendant allows the caller to enter the extension of the party that the caller wants to reach. Depending on the parameters established, the user can enter up to 16 digits from a touchtone telephone.

Automated Attendant is usually used by contact centers that do not have DID trunks and whose callers know the extension of the people they are calling. Because it reduces the need for “live attendants,” Automated Attendant reduces contact center costs.

The following example shows an example of a vector that implements Automated Attendant.

Example application - automated attendant

```
1. wait-time 0 seconds hearing ringback
     2. collect 5 digits after announcement 30001
        (‘‘You have reached Ridel Publications in Greenbrook.
        Please dial a 5-digit extension or wait for the
        attendant.’’)
     3. route-to digits with coverage y
     4. route-to number 0 with cov n if unconditionally
     5. stop
```

Step 1 of this vector contains the **`wait-time`** command, which is placed before the **`collect digits`** command in step 2 to provide the caller with ringback in the event that a TTR is not immediately available. A TTR must be connected in order for the **`collect digits`** command to take effect. Once a TTR is connected, the caller is prompted to enter the destination extension of the party he or she wants to reach (step 2). The **`collect digits`** command in step 2 collects the digits. Thereafter, the **`route-to digits`** command in step 3 attempts to route the call to the destination.

If the **`route-to digits`** command fails because the caller fails to enter any digits, or because the digits entered do not comprise a valid extension, then the **`route-to number`** command in step 4 routes the call to the attendant. However, as long as the destination is a valid extension, the **`route-to digits`** command succeeds, coverage applies, and vector processing terminates. Note that even if the destination is busy, vector processing terminates because coverage call processing takes effect.

Data in/voice answer and data/message collection example

This example involves a mutual fund company that is open 24 hours a day, 7 days a week. All incoming calls are directed to a single VDN extension that maps to a main vector. The main vector presents a menu of options to the calling party, and the vector also uses Call Prompting to determine the desired service. Three services are offered:

- New accounts enables the customer to open a new account.
- Account inquiries enables the customer to make inquiries concerning his or her account.
- Net asset values enables the customer to hear information concerning the net asset values of the company’s funds.

If the caller selects “account inquiries,” he or she is prompted to input his or her account number before being answered by an agent. The agent can use the CALLR-INFO button to display this number.

Note:

If the agent has a two-line display telephone, the account number is automatically displayed on the second line. Some supported display telephones include 6416, 6424, 8410, 8434 and CallMaster set.

This example uses three other applications that can be supported by the Call Prompting feature:

- Data In/Voice Answer (DIVA) allows a caller to receive information on a topic that he selects at the prompt. The caller selects the desired topic by entering the appropriate digits.
- Data Collection provides a method of collecting digits from a caller. The requested digits comprise an official number of some sort. For example, a Social Security Number, and they help the system process the call more efficiently.
- Message Collection allows the caller to leave a recorded message instead of waiting for the call to be answered.

The four vectors shown below illustrate how the mutual fund company handles telephone calls. Typically, the vector should be programmed to check if queue slots are available.

Example application - mutual fund company

```
VDN (extension=1030 name=”ABC Inv” vector=10 display override=”y”)
Vector 10
      1. wait-time 0 secs hearing ringback
      2. collect 1 digits after announcement 3531
         (‘‘Thank you for calling ABC Investments. If
         you wish to open a new account, please dial 1.  If
         you wish to make an account inquiry, please dial 2.
         If you wish to know the current net asset values of
         our funds, please dial 3.’’)
      3. route-to number 1031 with cov y if digit = 1
      4. route-to number 1032 with cov y if digit = 2
      5. route-to number 1033 with cov y if digit = 3
      6. route-to number 0 with cov n if unconditionally
      7. disconnect after announcement none

VDN (extension=1031 name=”New Account” vector=11)
Vector 11
      1. goto step 5 if calls-queued in split 1 > 19
      2. queue-to split 1 pri t
      3. announcement 3535
      4. wait-time 10 secs hearing music
      5. collect 1 digits after announcement 4020
         (‘‘We’re sorry.  All of our operators are busy at
         the moment.  If you’d like to leave your name and
         telephone number so that we can get back to you,
         dial 1.’’)
      6. goto step 10 if digit = 1
      7. announcement 3537
      8. wait time 50 secs hearing music
      9. goto step 6 if unconditionally
     10. messaging split 5 for extension 4000
     11. announcement 3538 (“We’re sorry, we cannot take
         your message at this time.  You may continue to hold, or
         you can call back later.’’)
     12. goto step 4 if unconditionally
```

DIVA and data/message collection vector examples (continued)

```
VDN (extension=1032 name=”Account Inq” vector=12)
Vector 12:
      1. wait-time 0 secs hearing ringback
      2. collect 6 digits after announcement 3533
         (“Please enter your 6-digit account number.”)
      3. goto step 7 if calls-queued in split 1 > 19
      4. queue-to split 1 pri m
      5. announcement 3535
      6. wait-time 60 secs hearing music
      7. collect 1 digits after announcement 4020
         (“We’re sorry. All of our operators are busy at
         the moment. If you’d like to leave your name and
         telephone number so that we can get back to you,
         dial 1.”)
      8. goto step 12 if digit = 1
      9. announcement 3537
     10. wait time 50 secs hearing music
     11. goto step 8 if unconditionally
     12. messaging split 5 for extension 4000
     13. announcement 3538 (“We’re sorry, we cannot take
         your message at this time. You may continue to hold, or
         you can call back later.”)
     14. goto step 4 if unconditionally

VDN (extension=1033 Name=”Net Asset Val” Vector=13)
Vector 13:
      1. disconnect after announcement 3534
         (‘‘The net asset values of our funds at the close
         of the market on Wednesday, May 15 were as follows:
         ABC Growth.....33.21.....up 33 cents; ABC
         High Yield.....11.48.....down 3 cents.’’)
```

When the call is placed, vector processing begins in vector 10, which is the main vector. Step 1 of the vector contains the **`wait-time`** command, which is placed before the **`collect digits`** command in step 2 to provide the caller with feedback in the event that a tone detector is not immediately available. Once a tone detector is connected, the **`collect digits`** command provides an announcement that requests the caller to enter **1**, **2**, or **3**, depending upon the service desired. If the caller enters a digit other than **1**, **2**, or **3** mentioned, or if the caller fails to enter any digits within 10 seconds, then the command fails and the call is routed to the attendant (step 6). If the caller enters **1**, **2**, or **3** within 10 seconds, then the call is routed to the vector specified in the appropriate **`route-to number`** command, which appears in steps 3, 4, and 5.

For instance, assume that, when prompted, the caller enters **3** because he or she wants to learn about the net asset values of the company’s funds. In such a case, the **`route-to number`** commands in step 3 and in step 4 fail, because in each case, the digit that is tested for in the condition portion of the command is not 3. However, the **`route-to number`** command in step 5 succeeds because the digit that is tested for matches the one entered by the caller. Accordingly, the call is routed to VDN extension 1033, and vector processing continues in vector 13.

The **announcement** command in step 1 of vector 13 provides the caller with the information on net asset values and then disconnects the call.

The process just described, whereby the caller receives information as a result of making a request at the prompt, is an example of the Data In/Voice Answer (DIVA) application.

Returning to the main vector, suppose that another caller wants to make an inquiry into his or her account, and the caller enters **2** when prompted. In such a case, step 3 fails, but step 4 succeeds. Accordingly, the call is routed to VDN extension 1032, and vector processing continues in vector 12.

The **collect digits** command in step 2 of vector 12 first requests the caller to enter his or her 6-digit account number. The command then collects the digits that are entered by the caller. Whether or not the caller correctly enters the digits, the **queue-to split** command in step 4 queues the call. If an agent does not immediately answer the call, the standard announcement is provided in step 5 and, if necessary, a delay is provided in step 6. The announcement in step 7 provides the caller with the option of leaving a message instead of having his or her call wait in queue. The caller is instructed to enter 1 if he or she wants to leave a recorded message. If the caller does not enter 1, the **goto step** command in step 8 fails, and an announcement-wait cycle is implemented by steps 9, 10, and 11 until the call is answered or abandoned. If the caller does enter 1 within 10 seconds, step 8 passes control to step 12. The **messaging split** command in step 12 attempts to connect the caller to an AUDIX or Message Center split so that the caller can leave a message. If the connection is made, the caller first hears ringback and can then leave a message. If the connection is not made, the step is unsuccessful, and step 13 provides an announcement that indicates that a connection could not be made. Thereafter, the **goto step** command in step 14 sends call control back to step 6, which leads the caller back into the steps to leave a message.

The process that was just described, whereby the caller, when prompted, enters digits that comprise an official number (an account number, in this case), is an example of the Data Collection application. If the agent has a CALLR-INFO button or a two-line display, the agent can see the digits that are entered by the caller. As a result, the agent need not request the account number from the caller.

Finally, suppose that a third caller wants to open an account and that he or she enters **1** when prompted in the main vector. In this case, step 3 of the main vector is successful. Accordingly, the call is routed to VDN extension 1031, and vector processing continues in vector 11.

In step 2 of vector 11, the call is queued to the main split. Thereafter, if necessary, step 3 provides the appropriate announcement, and step 4 provides a delay period. The announcement in step 5 provides the caller with the option of leaving a recorded message instead of having his or her call wait in queue. This is an example of the Message Collection application. The caller is instructed to enter **`1`** if he or she wants to leave a recorded message. If the caller does not enter **`1`**, the **`goto step`** command in step 6 fails, and an announcement-wait cycle is implemented by steps 7, 8, and 9 until the call is answered or abandoned. If the caller does enter **`1`** within 10 seconds, step 6 passes control to step 10. The **`messaging split`** command in step 10 attempts to connect the caller to an AUDIX or Message Center split so that the caller can leave a message. If the connection is made, the caller first hears ringback and can then leave a message. If the connection is not made, the step is unsuccessful, and step 11 provides an announcement that indicates that a connection could not be made. Thereafter, the **goto step** command in step 12 sends call control back to step 4, which leads the caller back into the steps to leave a message.

Distributed contact centers example

This example involves two customer contact centers located in New York and Denver. Calls to the New York contact center are queued to up to two splits. If calls remain unanswered for a period of time, a Look-Ahead Interflow (LAI) call attempt is made to the Denver contact center. If there are 10 or fewer queued calls in Denver, the LAI call attempt is accepted and serviced there. Otherwise, the call is denied and remains in queue in New York until an agent becomes available. The two vectors shown below illustrate the process.

Note:

For other examples of LAI, see Look-Ahead Interflow (LAI) on page 203. To learn how to integrate distributed contact centers using multi-site Best Service Routing, see Best Service Routing (BSR) on page 229.

Example application - distributed call centers

```
SENDING SWITCH:
VDN (extension=1080  name=‘‘New York Office’’  vector=80)
Vector 80:
       1. goto step 11 if calls-queued in split 1 pri m > 5
       2. queue-to split 1 pri m
       3. announcement 3580 (‘‘All of our agents
          are busy.  Please hold and you will be answered
          by the first available agent.’’)
       4. wait-time 6 seconds hearing music
       5. route-to number 913035661081 with cov n if unconditionally
       6. check split 2 pri m if calls-queued < 5
       7. wait-time 6 seconds hearing music
       8. announcement 3581 (‘‘All of our agents
          are still busy.  Please hold and you will be
          serviced by the first available agent.’’)
       9. wait-time 60 seconds hearing music
      10. goto step 5 if unconditionally
      11. busy
RECEIVING SWITCH:
VDN (extension=1081 Name=‘‘Denver Inflow’’ Vector=81)
Vector 81:
       1. goto step 7 if calls-queued in split 3 pri l > 10
       2. wait-time 0 seconds hearing music
       3. queue-to split 3 pri h
       4. announcement 3582 (‘‘We apologize
          for the delay.  Please hold and you will be
          serviced by the first available agent.’’)
       5. wait-time 60 seconds hearing music
       6. goto step 5 if unconditionally
       7. disconnect after announcement none
```

In this example, vector 80 is on the sending switch from a contact center in New York, while vector 81 is on the receiving switch at a contact center in Denver.

In the sending switch, the call is queued to split 1 at a medium priority (step 2) if the condition in step 1 is met. If the condition is not met, the call is routed to busy in step 11.

If the call is queued but not immediately answered, an announcement (step 3) and music (step 4) are provided. If the call is still not answered at this point, step 5 places a LAI call attempt to the receiving switch, on which vector 81 resides.

Step 1 in the receiving switch determines whether the call can be serviced in Denver. If the number of calls queued at any priority in split 3 is greater than 10, vector 81 cannot service the call. In such a case, control is passed to step 7, which rejects the Look-Ahead Interflow call attempt. However, if the test in step 1 succeeds, the call is queued by the receiving switch in split 3 at a high priority (step 3) and the LAI call attempt is accepted. Accordingly, the call is removed from the main split queue in New York, and control is passed to the Denver switch, where vector processing continues at step 4.

If the receiving switch does not accept the LAI call attempt, control is passed to step 6 of the sending vector. This step then queues the call to split 2 at a medium priority, provided that there are fewer than five calls queued in that split. Thereafter, the customary announcement-wait sequence is implemented (steps 7, 8, and 9). Finally, if necessary, step 10 sends control back to step 5, which makes another LAI attempt, and the cycle is repeated.

Note:

To avoid confusing the caller, the treatment provided at the receiving switch should be consistent with the treatment that is provided at the sending switch. In the distributed contact centers example, note that the caller hears music (and never ringback or silence) at the sending switch. Accordingly, music should be (and, in our example, is) featured at the receiving switch.

Help desk example

This example involves a help desk at a computer firm. The help desk is configured into three groups. One group handles hardware problems, the second group handles software problems, and the third group handles general problems. For this application, the information that is provided in the ASAI Route request, that is, calling party number, called number, collected digits, is used to route the call to the most appropriate agent. Such an agent might be the one who last serviced the caller, or it might be the next available agent for the specific caller. Also, based on switch traffic conditions and the caller-entered digit, the call can be diverted to other destinations, such as other ACD splits, announcements, or switches.

The following vector shows the help desk application.

Example application - help desk

```
1. collect 1 digits after announcement 4704
        (‘‘Welcome to the TidyBits Computer Corporation help desk.
        If you have a question about hardware, please dial 1.
        If you have a question about software, please dial 2.
        If you have a general question, please dial 3.’’)
     2. adjunct routing link 2400
     3. wait-time 4 seconds hearing ringback
     4. route-to number 3710 with cov y if digit = 1
     5. route-to number 3720 with cov y if digit = 2
     6. route-to number 3730 with cov y if digit = 3
     7. route-to number 0 with cov n if unconditionally
     8. stop
```

In step 1 of this vector, the caller is instructed to enter **1**, **2**, or **3**, depending upon the service (hardware, software, general) that he or she desires. Thereafter, the **adjunct routing** command in step 2 instructs the switch to send a Route request to the adjunct processor, which is connected to extension 2400. The Route request contains the called party number, the calling party number, and the digit that is collected in step 1, along with the other pertinent information for adjunct routing (see Adjunct (ASAI) Routing on page 163). If **1**, **2**, or **3** is not entered, and if the adjunct does not return a route, the call is eventually routed to the attendant (step 7).

If the **adjunct routing** command in step 2 succeeds, the adjunct uses the information included in the Route request to select the appropriate route for the call. Let's assume the caller enters **1** and the **adjunct routing** command succeeds. In such a case, if the caller is judged to be a "prime" hardware customer, the call might be routed to one of a handful of specific agents who are assigned to handle such customers. On the other hand, if the caller is judged to be a "casual" hardware customer, the call might be routed to a larger group of ACD agents before it is queued, or to an appropriate announcement.

Finally, assume that the caller enters **1** and that the **adjunct routing** command fails. In such a case, the call is routed by the **route-to number** command in step 4, probably to a vector that queues the call or provides an appropriate announcement.

Insurance agency/service agency example

This example involves an insurance company contact center. It handles calls from independent field agents, policy holders with claims, policy holders needing customer service, and several general service agency type 800 number client accounts. Each different type of call has its own 800 number that routes the calls to associated VDNs.

The following list describes the contact center requirements.

- The independent field agents require fast service. They call the company to find out the latest rates for specific clients, to set up policies, to make adjustments, and so on. Often their clients are waiting as they call. Therefore the insurance company wants to maintain an Average Speed of Answer (rolling-ASA) of 30 seconds or less for field agent calls. These are the most important calls and are given high priority in queues.
- The calls to claims must be separated by area code. The claims agents receive different training based on the area of the country for the claim. A particular group of agents can be given training for more than one area code. Therefore, area codes do not need to be tested individually and can be grouped in vector routing tables.
- The insurance company wants to give customer service callers an announcement indicating how long that they can expect to wait for service.
- The insurance agency is also selling spare contact center capacity to client accounts. The account contracts are provided on the basis that only so many calls to a particular account are accepted at any given time.

In this example, rolling ASA Routing is used to maintain the rolling ASA objective of 30 seconds or less for field agent calls. ANI Routing is used to partition calls based on area code and route the calls to the appropriate claims agents. EWT Routing is used to notify customer service callers of their expected wait time if it is longer than 60 seconds. VDN Calls Routing is used to regulate the number of calls to service agency clients.

The following table shows the VDNs and vectors that are associated with each type of call.

VDN table for insurance agency or service agency example

Type of service	VDN number	Vector number
Field Agents	1001	1
Claims	1002	2
Customer Service	1003	3
Client 1	1004	4
Client 2	1005	5

Note:

To more clearly demonstrate the features described in this example, the sample vectors do not include tests for unstaffed or full queues, out-of-hours operation and so forth.

An example of a vector that could be used to maintain a rolling ASA of 30 seconds for field agent calls is shown in the following figure.

Field agent vector example

Step 1 queues the call to the main split. If the main split is currently answering calls within the target time of 30 seconds, step 2 bypasses all of the backup splits and goes directly to the announcement in step 6. The assumption is that the call will be handled by split 10 within the time constraints. However, if the call is not answered by the time that vector processing reaches step 8, the backup splits are checked.

If the rolling ASA for the main split is greater than 30 seconds, steps 3, 4, and 5 check backup splits. The call is queued to any of these splits that have a rolling ASA of 30 seconds or less. If the call still is not answered by the time vector processing reaches step 8, then the backup splits are checked again.

The following vector example could be used to route claims calls by area code.

Claims vector example

```
VDN 1002 -- Claims Calls

     1. goto step 10 if ani = none
     2. goto vector 21 if ani = 201+
     3. goto vector 22 if ani = 212+
     4. goto vector 23 if ani in table 1
     5. goto vector 24 if ani in table 2
     6. goto vector 25 if ani in table 3
     7. goto vector 26 if ani in table 4
     8. goto vector 27 if ani in table 5
     9. goto vector 30 if unconditionally
    10. wait-time 0 seconds hearing ringback
    11. collect 3 digits after announcement 10001
        (“Please dial your area code”)
    12. goto vector 30 if digits = none
    13. goto vector 21 if digits = 201+
    14. goto vector 22 if digits = 212+
    15. goto vector 23 if digits in table 1
    16. goto vector 24 if digits in table 2
    17. goto vector 25 if digits in table 3
    18. goto vector 26 if digits in table 4
    19. goto vector 27 if digits in table 5
    20. goto vector 30 if unconditionally
```

Each vector routing table referenced in the example shown above contains a list of area codes with the “+” wildcard. Each list of area codes is handled by a specific group of agents. Vectors 21 through 27 queue calls to the appropriate group of agents. Vector 30 provides a live agent to screen calls that have area codes that are not listed in any table or vector step. It also provides access to an agent when ANI is not available and the caller did not enter an area code when prompted.

The following vector example notifies customer service callers of their expected wait time unless they will not have long to wait.

Customer service vector example

```
VDN 1003 -- Customer Service Calls

     1. goto step 10 if expected-wait for split 32 pri l > 600
     2. queue-to split 32 pri l
     3. wait-time 20 seconds hearing ringback
     4. goto step 8 if expected-wait for call > 40
     5. announcement 1100
     6. wait-time 40 seconds hearing music
     7. goto step 5 if unconditionally
     8. converse-on split 80 pri l passing wait and none
     9. goto step 5 if unconditionally
    10. disconnect after announcement 1400
```

In step 1, callers who would wait more than 10 minutes are routed to a “call back later announcement.” Step 4 routes callers to a Conversant VRU to be given the expected wait time announcement while they hold their place in the queue.

The following vector examples can be used to regulate the number of calls to service agency clients. In this example, Client 1 has contracted for 100 simultaneous calls while client 2 has contracted for only 50 simultaneous calls.

Service Agency Clients Vectors examples

```
VDN 1004-- Client 1 Calls

     1. goto step 3 if counted-calls to vdn 1004 <= 100
     2. busy
     3. queue-to split 60 pri l
     4. wait-time 20 seconds hearing ringback
     5. announcement 12000
     6. wait-time 60 seconds hearing music
     7. goto step 5 unconditionally


     VDN 1005 -- Client 2 Calls
     1. goto step 3 if counted-calls to vdn 1005 <= 50
     2. busy
     3. queue-to split 60 pri l
     4. wait-time 20 seconds hearing ringback
     5. announcement 12000
     6. wait-time 60 seconds hearing music
     7. goto step 5 unconditionally
```

In both of the example vectors shown above, the first step routes calls to queue if the number of contracted calls is not exceeded. Otherwise callers receive a busy signal.

Warranty service (with EAS) example

This example involves a major appliance company that offers one year warranties and extended warranties on its major appliances, such as dishwashers, refrigerators, washers, and dryers. The warranties are printed in English and Spanish to accommodate customers who speak and understand these languages. Naturally, callers need to speak with someone who is familiar with the appliances they have bought and who speaks the appropriate language. Accordingly, 800 numbers are provided for calling both English-speaking agents and Spanish-speaking agents. Bilingual agents with Spanish-speaking skills are hired so that they can back up the groups of English-speaking agents. Agents are trained first on all appliance models of a certain type and then on all appliance models for a room, such as the kitchen, the laundry room, and so forth.

The skills shown in the following table are required for the warranty service contact center:

Skill table for a warranty service contact center

Appliance type	English skill number	Spanish skill number
Kitchen appliances	10	20
Dishwashers	11	21
Refrigerators	12	22
Laundry appliances	30	40
Washers	31	41
Dryers	32	42
Supervisors		100

The VDN Skill Preferences are set up as shown in the following table.

VDN skill table for the warranty service contact center

VDN skill preference	Appliance	VDN	First	Second	Third
English	Dishwasher	1100	11	10	20
	Refrigerator	1101	12	10	20
	Washer	1102	31	30	40
	Dryer	1103	32	30	40
Spanish	Dishwasher	1200	21	20	--
	Refrigerator	1201	22	20	--
	Washer	1203	41	40	--
	Dryer	1204	42	40	--

The agent skills are set up as shown in the following table.

Agent skills for the warranty service contact center

Agent	Skill level 1		Skill level 2	
Kim	42	40	41	30
Michelle	100	--	--	--
Beth	31	--	--	--
Mike	32	--	30	--

Once skills are assigned to VDNs and to agents, calls are directed to the appropriate vector.

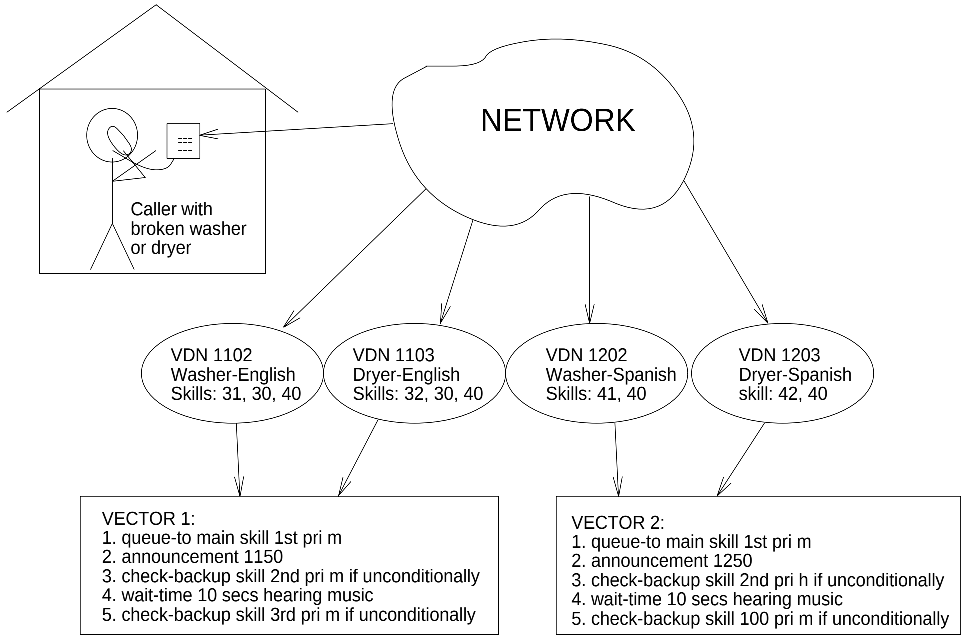
The goal of the warranty service contact center is to answer 80% of the incoming calls within 20 seconds. Accordingly, if a call that is directed to a vector is not answered by the time the announcement finishes, a second group of agents is viewed, thus enlarging the agent pool. If the call is not answered within the following 10 seconds, a third group of agents is viewed.

Since the contact center has only a few bilingual agents, the center’s management wants to reserve these agents for Spanish-speaking callers. This can be done by giving Spanish-speaking callers a higher priority in the vector or by assigning a higher skill level to Spanish skills. Also, if a Spanish-speaking caller waits more than 30 seconds for service, a supervisor of the Spanish-speaking skills takes the calls.

Warranty service contact center (part 1) and Warranty service contact center (part 2) show the setup for the warranty service call service. Specifically, the figures show the vectors and call flows for callers with a broken washer or dryer who need service. Separate vectors are used to provide an announcement in Spanish and in English (see step 2). The same two vectors can be used for callers who need service for broken dishwashers and refrigerators.

The following figure shows how the call comes into the network and is then directed to the appropriate VDN, which in turn points to the appropriate vector. For each VDN, the corresponding VDN skills are indicated.

Warranty service contact center (part 1)

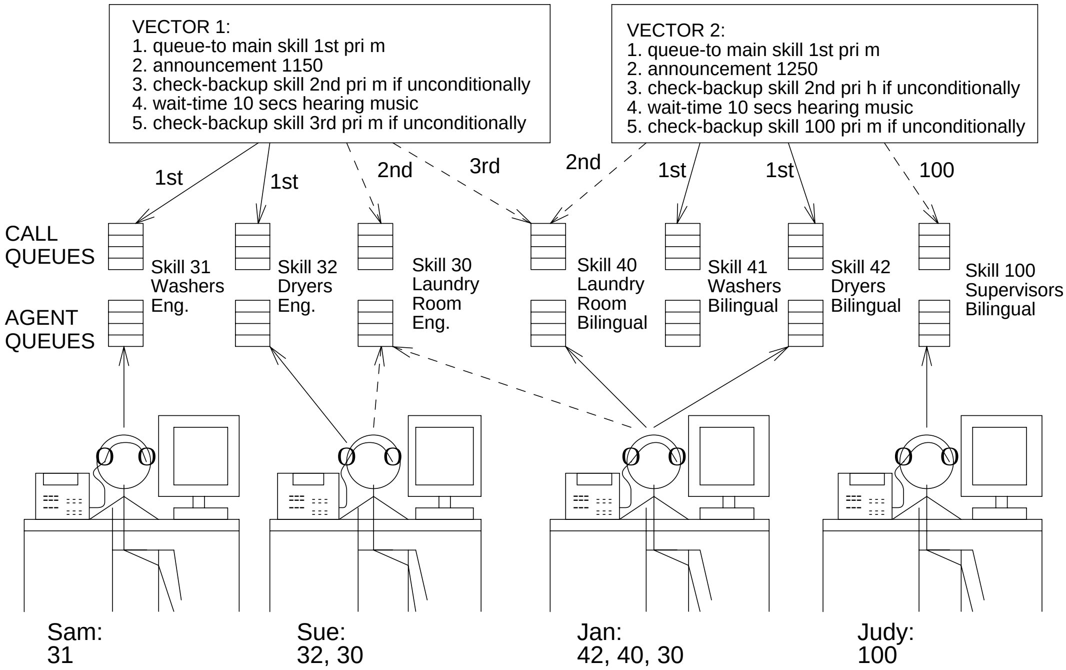


The next figure shows how the vector-processed call is directed to the appropriate call queue. The figure also shows how the call is directed to the appropriate agent or agents. The agent skills are indicated below each agent's name. Dashed lines indicate backup or secondary skills.

Note:

Only a small sample of agents is shown in the example figure.

Warranty service contact center (part 2)



Assume that a Spanish-speaking caller has a broken dryer and decides to call the warranty service contact center. The caller dials the appropriate number. The call then enters the switch and is directed to VDN 1203, which points to Vector 2. As illustrated earlier, VDN skill preferences 42 (dryers) and 40 (laundry appliances) are administered as the 1st and 2nd skill preferences, respectively, for VDN 1203.

Once vector processing starts, the **`queue-to skill`** command in step 1 of Vector 2 queues the call to the skill group corresponding to the first VDN skill (42-Dryers Bilingual). If an agent with skill 42 (Jan, for example) is available, this agent answers the call. If such an agent is not available, the appropriate delay announcement in step 2 is played. Next, the **`check skill`** command in step 3 attempts to queue the call to the skill group corresponding to the 2nd VDN skill (40-Laundry Appliances Bilingual). If an agent with skill 40 is available (Jan, for example), that particular agent answers the call. Otherwise, a wait period is provided in step 4, and the **`check skill`** command in step 5 checks the “specific” skill (100-Supervisors Bilingual) for available agents.

Resort reservation service (with EAS) example

This example involves a resort company that places a variety of advertisements in magazines for information on a particular resort or state. Callers respond to these advertisements dial one of several numbers provided in the advertisement. A contact center makes the reservations for the resort company. To satisfy the request of many callers to the service, an effort is made to have callers connected to an agent who has visited the resort they are interested in visiting. Also, the resort company has determined that it is easier to sell additional sightseeing packages if the agent has a regional accent.

Placing the reservation

To respond to an advertisement, the caller can dial a number that directly routes him or her to a VDN for that state's resorts. As an alternative, the caller can dial the general number for the resort chain and be serviced using the Call Prompting feature. The following sections discuss these methods.

Specific number dialing

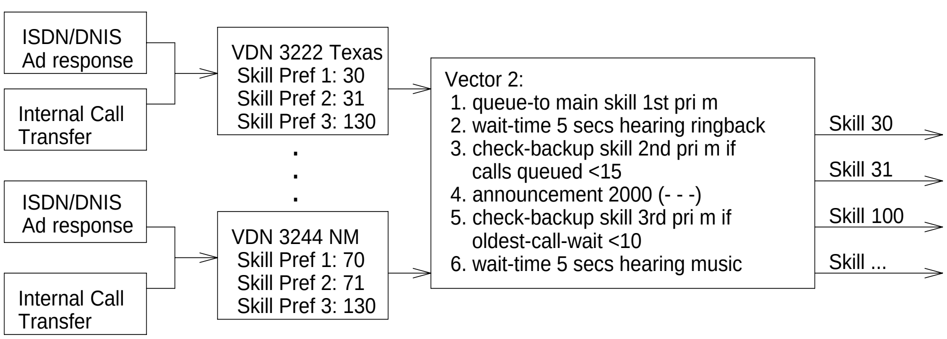
The contact center is set up in such a way that a VDN with an accompanying set of VDN Skill Preferences is assigned to each state that has a resort. For example, the following table shows how Skill Preferences are assigned to Texas VDN 3222.

VDN 3222 skill preferences assignments for the resort reservation service

Texas VDN 3222 skill preferences		
Skill preference	**Skill number**	**Agent skill**
1st:	30	Agent who has a Texas accent and has visited resorts in Texas
2nd:	31	Agent who has visited resorts in Texas
3rd:	130	Any agent who can take a reservation

The following figure shows how a call to VDN 3222 can be processed by Call Vectoring.

Process involving specific number dialing



For this process, a single VDN for each state is assigned to Vector 2. Accordingly, the figure shown above shows the VDN and the associated VDN skills for two states, Texas and New Mexico.

Assume that a caller wants information on resorts in Texas and dials the appropriate number, for example, 615-3222. In this case, the call enters the switch and is directed to VDN 3222, which points to Vector 2.

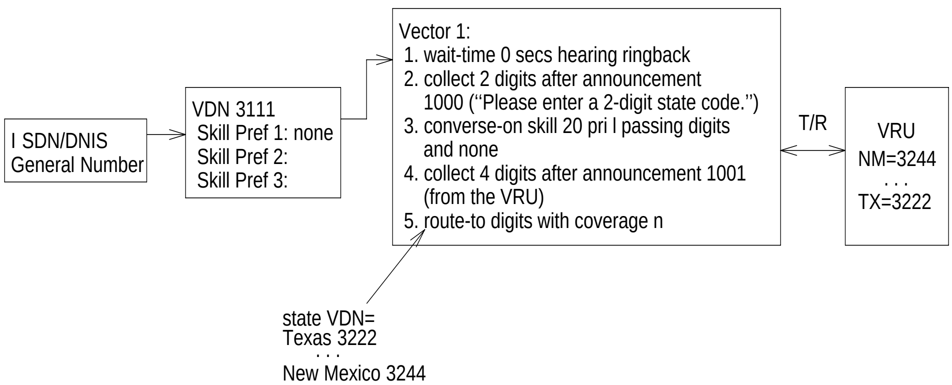
Once vector processing starts, the **queue-to skill** command in step 1 queues the call to the skill group that corresponds to the 1st VDN skill (30-Agent with a Texas accent who has visited resorts in Texas). If an agent with skill 30 is available, this agent answers the call. If such an agent is not available, the **check skill** command in step 3 attempts to queue the call according to the stated conditions (if calls-queued < 15) to the skill group that corresponds to the 2nd VDN skill (31-Agent who has visited resorts in Texas). If step 3 fails, the **check skill** command in step 5 attempts to queue the call vis-a-vis the stated conditions (if the oldest-call waiting < 10) to the skill group that corresponds to the 3rd VDN skill (100-Any agent who can take a reservation).

General number dialing

This option allows the caller to dial the general number provided, for example, 615-3111. The caller is then serviced in part using the Call Prompting feature.

The following figure shows how a call to VDN 3111 can be processed using Call Vectoring.

Figure 4-1: Process involving general number dialing



After the number is dialed, the call is directed to VDN 3111, which points to Vector 1. Note there are no skill preferences assigned to VDN 3111. Also, VDN 3111 is the only VDN that is administered to point to Vector 1. Therefore, this VDN is used for calls from all states.

The **collect digits** command in step 2 of the previous vector first requests the caller to enter the appropriate 2-digit state code and then collects the digits. Assume that the caller enters the correct code for Texas, which is "05." In this case, the **converse-on skill** command in step 3 delivers the call to the converse skill if there is a queue for the skill and the queue is not full, or if a VRU port is available.

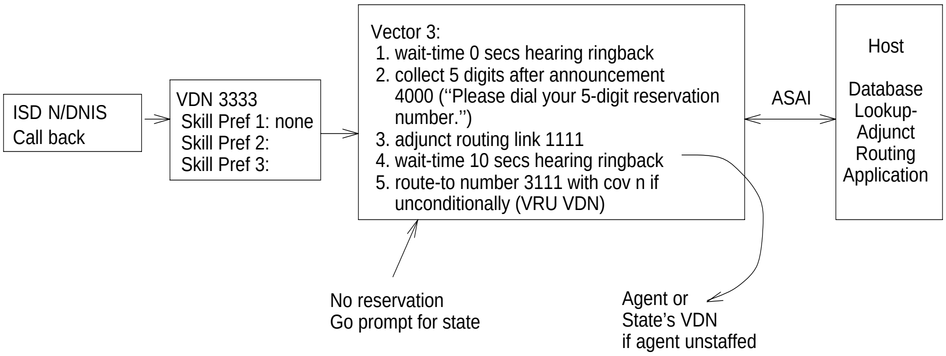
For more information about the **converse-on** command, see Chapter 5: Basic Call Vectoring on page 105.

When the VRU port responds, the step then outpulses the state code "05" to the VRU using the **passing digits** parameter that is included in the command. Once the VRU receives this state code, the VRU in turn outpulses the Texas VDN (3222) to the switch. Thereafter, the **collect digits** command in step 4 collects the digits that comprise this VDN. Finally, the **route-to digits** command in step 5 routes the call to Texas VDN 3222, which points to Vector 2. This process is discussed in the General number dialing section.

Call-back provisions

After a caller makes a reservation for a resort site, the caller is given a call-back number. Such a number is helpful if the caller needs more information or wants to check on some arrangement that was previously made. The following figure shows one approach for enabling call-back provisions.

Example 8: Call-back provisions



After the number is dialed, the call is directed to VDN 3333, which points to Vector 3. Note that there are no skill Preferences assigned to VDN 3333. Also, VDN 3333 is the only VDN that is administered to point to Vector 3. Therefore, this VDN is used for calls from all states.

The **`collect digits`** command in step 2 of the previous vector first requests the caller to enter his or her 5-digit reservation number and then collects the digits. Once the digits are collected, the **`adjunct routing`** command (if successful) in step 3 causes the switch to send the collected digits (along with other information) to the host in the ASAI adjunct routing request. The host then uses these digits to perform a database lookup for the agent who made the reservation and the resort that corresponds to the reservation. If the agent is currently logged in, the call is automatically routed to the agent. Once this happens, information on the relevant reservation is displayed at the agent's data terminal, thus providing quicker and more personal service. If the agent is not logged in, the call is routed to step 5, where the **`route to`** command unconditionally routes the call to the VRU VDN 3111. This process is discussed in the General number dialing section.

Attendant routing example

The following example show how the Attendant Vectoring commands can be used to route calls in an attendant environment. For the attendant vectors, consider the following vectors and vector administration.

Note:
For the following vector examples, tenant partitioning is turned on:

Attendant Vectoring vectors

VDN 1999 **vector 1**	**VDN 2999** **vector 2**	**VDN 3999** **vector 3**
1. wait-time 0 secs hearing ringback 2. goto step 6 if time-of-day is all 12:00 to 13:00 3. queue-to attd-group 4. goto step 7 if queue-fail 5. wait 999 secs hearing music 6. busy 7. route-to number 4000 with cov y if unconditionally 8. route-to number 93035381000 with cov y if unconditionally	1. wait-time 0 secs hearing ringback 2. queue-to attd-group 3. goto step 6 if queue-fail 4. announcement 9000 5. wait 999 seconds hearing music 6. disconnect after announcement 9001 7. queue-to hunt-group 1 8. goto step10 if queue-fail 9.wait 999 secs hearing ringback 10. busy 11. route-to number 93035381000 with cov y if unconditionally	1. wait-time 0 secs hearing ringback 2. goto step 7 if time-of-day is all 12:00 to 13:00 3. queue-to attd-group 4. goto step 7 if queue-fail 5. announcement 9000 6. wait 15 seconds hearing music 7. goto step 4 if unconditionally 8. queue-to attendant 6000 9. goto step 10 if queue-fail 10. wait 999 secs hearing ringback 11. route-to number 93035381000 with cov y if unconditionally

Vector administration

- All stations are assigned TN 1 which is associated with attendant group 1, VDN 1999, and music source 1.
- All trunk groups are assigned TN 2 which is associated with attendant group 1, VDN 2999, and music source 2.
- All VDNs are assigned TN 3 which is associated with attendant group 2, VDN 3999, and music source 3.
- Extension 4000 is assigned to a hunt group 1.
- Extension 6000 is assigned to an attendant console for direct access.

Local attendant group access code

When a station dials the attendant access code, the call is redirected to vector 1. If it is lunch time, the call is sent to a hunt group and vector processing terminates. If it is not lunch time, the call is sent to attendant group 1. If an attendant is available, the call is terminated to the attendant and vector processing terminates. Otherwise, the call is queued to the attendant group and the caller hears music from the music source that is assigned to TN 1 until an attendant answers the call. If the call cannot be queued, it is routed to a remote location with coverage, and vector processing terminates. If the call is unanswered after 999 seconds in the attendant queue, the caller hears a busy signal and vector processing terminates.

Note:

The **`route-to`** command leaves vector processing as soon as the call is successfully routed. So, in the example above, if it is lunch time the call will route to the hunt group and all hunt group processing will then apply. If the group is assigned a queue, the call is queued. If the group is not assigned a queue and the coverage criteria is met, the call follows the hunt group's coverage path. If the hunt group is in night service, the call goes to the hunt group's night service destination. If the route-to command indicates coverage **n**, the hunt group's coverage path is not followed and vector step 7 applies.

Incoming trunk calls to attendant group

When a call is received on a trunk that has the attendant group assigned as the incoming destination or when the call is addressed to the attendant group, the call is redirected to vector 2. The call is then sent to attendant group 1. If an attendant is available, the call is terminated to the attendant and vector processing terminates. Otherwise, the call is queued to the attendant group and the caller hears the announcement followed by music from the music source that is assigned to TN 2. If the call is unanswered after 999 seconds in the attendant queue, the caller is dropped after hearing an announcement and vector processing terminates. If queueing to the attendant fails, the call is queued to hunt group 1. If a member is available to take the call, the call is terminated to the member and vector processing terminates. If a member is not available and the call can be queued, the call is queued and the caller hears ringback until a member answers. If the call is unanswered after 999 seconds in the hunt group queue, the caller hears busy and vector processing terminates. If the call cannot be queued, the call is routed to the remote location and vector processing terminates.

Note:

The main difference from the example shown in Local attendant group access code on page 84 is queueing the call to the hunt group rather than routing the call there. In this example, the call will not follow the hunt group's coverage path or night service destination.

Incoming LDN calls

When a call is received for an LDN, the call is redirected to vector 3. If it is lunch time, the call is sent to attendant 6000. If the attendant is available, the call is answered and vector processing terminates. If the attendant is not available, the call is placed into queue and the caller hears ringback until the attendant answers the call. If the call is unanswered after 999 seconds in the attendant's queue, the call is sent to the remote location and vector processing terminates. If the call cannot be placed in attendant 6000's queue, the call is routed to a remote location and vector processing terminates. If it is not lunch time, the call is sent to attendant group 2. If an attendant is available, the call is terminated to the attendant and vector processing terminates. Otherwise, the call is queued to the attendant group and the caller hears an announcement followed by music from the music source assigned to TN 3 every 15 seconds. If the call cannot be queued, it is sent to attendant 6000.

Note:

Vector 3 attempts to queue the call to attendant 6000. A **`route-to`** command could also be used, but care should be taken since an attendant cannot be assigned a coverage path.

QSIG CAS example

This example shows how you can use Attendant Vectoring with CAS.

CAS branch

Suppose the contact center always wants to play an announcement at a QSIG CAS branch before routing the call to the QSIG CAS main. In this case, assume that an attendant VDN needs to be administered in the QSIG CAS Number field at the branch instead of the number to the QSIG CAS main attendant access code, which is 303-538-0 with an AAR access code of 9 in this example. The following vector plays an announcement and then routes the call to the QSIG CAS main.

Administration for vector 1 of the attendant VDN is shown in the following Call Vector example.

QSIG CAS vector main

```
change vector 1                                               Page 1 of 3
                    CALL VECTOR

    Number: 1        Name: Night station service vector 4
Multimedia? n     Attendant Vectoring? y     Lock? y
     Basic? n  EAS? n  G3V4 Enhanced? n  ANI/II-Digits? n  ASAI Routing? n
 Prompting? y  LAI? n G3V4 Adv Route? n  CINFO? n  BSR? n   Holidays? n

01 announcement 9000
02 route-to number     93035380       with cov y if unconditionally
03
04
05
06
07
08
09
10
11
```

CAS main

Calls from a QSIG branch are sent to the QSIG CAS main with the main attendant access code as the destination address. Therefore, these calls automatically become attendant group calls. The VDN to which these calls are redirected depends on the TN of the incoming trunk.

Night station service example

This example shows how you can use the Attendant Vectoring features for night service.

Night station service vectors 4 and 5

```
change vector 4                                               Page 1 of 3
                    CALL VECTOR

    Number: 4        Name: Night station service vector 4
Multimedia? n     Attendant Vectoring? y     Lock? y
     Basic? n  EAS? n  G3V4 Enhanced? n  ANI/II-Digits? n  ASAI Routing? n
 Prompting? y  LAI? n G3V4 Adv Route? n  CINFO? n  BSR? n   Holidays? n

01 route-to     number 9303538100       with cov y if unconditionally
02
03
04
05
06
07
08
09
10
11

change vector 5                                               Page 1 of 3
                    CALL VECTOR

    Number: 5        Name: Night station service vector 4
Multimedia? n     Attendant Vectoring? y     Lock? y
     Basic? n  EAS? n  G3V4 Enhanced? n  ANI/II-Digits? n  ASAI Routing? n
 Prompting? y  LAI? n G3V4 Adv Route? n  CINFO? n  BSR? n   Holidays? n

01 route-to     number 6000        with cov n if unconditionally
02 route-to     number 93035381000 with cov y if unconditionally
03
04
05
06
07
08
09
10
11
```

Administration for vector 4 and vector 5 of VDN 4999 is as follows.

- Trunk group 1 is assigned TN 2 which is associated with attendant group 1, and night destination 4999.
- Trunk group 2 is assigned TN 1 which is associated with attendant group 2, and night destination 5999.
- Extension 6000 is assigned to a station.
- System night service is on.

When a non-DID call comes in on trunk group 1, the call is redirected to VDN 4999 which routes it to a remote location.

When a non-DID call comes in on trunk group 2, the call is redirected to VDN 5999 which routes it to station 6000. If station 6000 is unavailable, the call does not cover on station 6000’s coverage path. Vector processing continues and routes the call to a remote location.

Note:

When station night service is active, calls are processed according to the administered night destination for the trunk group, not the night destination for the associated TN. In other words, these are not attendant group calls. If the night destination is assigned as attd or left unassigned, the calls become attendant group calls and are processed according to the partitions night destination.

Holiday Vectoring example

This example is a vector that is directing calls to special processing because of a holiday or special event. Holiday Vectoring is an enhancement that simplifies vector writing for holidays. It is designed for customers who need to reroute or provide special handling for date-related calls on a regular basis.

In this example, a commercial bank that is headquartered in Germany has branches in Europe. The bank recently established a U.S. presence by opening branches in the New York City metropolitan area. The bank's credit card division operates two 100-agent contact centers in Ireland and Germany and one 50-agent contact center in the U.S.

All agents in the European centers are bilingual and assigned to splits that handle calls from both English and German customers. The same is true for the agents in the New York contact center. Because the New York contact center is open 24 hours a day, it often takes calls that are routed from the Irish and German contact centers after those centers close at 6:00 p.m. local time.

Due to the large number of bank holidays per year in Europe (up to 30 days), the Holiday Vectoring feature can be used to create vectors that distribute calls automatically on holidays. The contact center administrator recommended this feature to the systems administrator to save the cost of time spent on writing vectors for date-related processing, and to save business that would be lost to abandoned calls if vectors are not readministered for holidays.

The following figure indicates that, beginning on December 24 and continuing through 6:00 am on January 2, incoming calls to the contact center in Germany will be processed as Christmas holiday calls.

Note:

Because date ranges must be within the same calendar year, New Year's Day had to be entered as a separate item.

Setting up a holiday table

```
change holiday-table 1                                          page 1 of 1
                                HOLIDAY TABLE

    Number: 1              Name: Bank Holidays

          START                   END
    Month Day Hour Min     Month Day Hour Min      Description
    12    24               12    31                Christmas
    01    01  00   00      01    02  06   00       New Year's Day
```

After submitting the Holiday Tables form, the next step is to modify the vector processing for these holidays. On the Call Vector form, enter the new goto conditional for the holidays.

Modifying a vector to route according to a holiday table

```
change vector 3                                                   Page 1 of 3
                    CALL VECTOR

    Number: 3        Name: In Ireland
Multimedia? n     Attendant Vectoring? n     Lock? y
     Basic? y  EAS? n  G3V4 Enhanced? n  ANI/II-Digits? n  ASAI Routing? n
 Prompting? y  LAI? n G3V4 Adv Route? n  CINFO? n  BSR? n   Holidays? y

01 goto         vector   2  if holiday         in    table 1
02 route-to     number 123456789       with cov n if unconditionally
```

The setup for the vector routes the call to the United States contact center. For example, if someone in Europe calls the bank before 6:00 a.m. on January 2, the call is routed to the United States contact center. If someone in Europe calls after 6:00 a.m. on January 2, the call is routed to the German contact center.

Network Call Redirection example

This example shows the primary, status poll, and interflow vectors that redirect calls on the public network using the NCR feature.

Note:

This example assumes knowledge of multi-site BSR applications. For information about BSR, see Best Service Routing (BSR) on page 229 For information about NCR, see Network Call Redirection on page 283.

The e-Commerce company used in this example has three contact centers. In an effort to reduce costs, the company has implemented Network Call Redirection (NCR) to redirect calls on the public network and reduce the trunking costs between the three switches. BSR is also implemented on the switches in order to increase the efficiency of agent utilization.

The e-Commerce company receives calls from a public network. Trunks used to deliver calls from the public network have been assigned Network Call Transfer (NCT) capabilities. NCT occurs after the incoming call is initially answered. With NCT, the switch is required to set up the second leg of the call and then wait for the second site to acknowledge before requesting the public network to transfer the first leg of the call to the second leg, and before the public network drops the trunks to the switch. The benefit is that the switch retains control over the call and can redirect the call using the trunk-to-trunk method should the NCT invocation fail.

After the second leg of the call is initiated and acknowledged by the public switch, the public network joins the original ISDN caller to the redirected-to endpoint and then drops both the original ISDN call and the second leg of the call at the redirecting switch.

To activate the NCR feature for each site, the switch Administrator ensures that the `Net Redir` field on the BSR Application Table form is set to `y` for the location entry.

The e-Commerce company has set up IP trunking to emulate ISDN PRI and will use this capability to poll remote sites for possible NCR. For information on setting up IP trunking to emulate ISDN PRI, see the *Administration for Network Connectivity for Avaya MultiVantage Software*, 555-233-504.

The following sections give examples of how the vectors must be set up at each site to use the public network via NCR (as opposed to IP trunking) to route a call from one site to another. For information about administering BSR polling over IP, see “Administering BSR IP polling without B-Channel” in the “Best Service Routing” chapter of *Avaya MultiVantage Call Center Software – Guide to ACD Call Center*, 555-230-716.

Primary Vector

A call arrives at eCommerce location 1 and is processed by the primary vector. This vector begins the BSR process by considering the specified resources. The following Call Vector example shows the primary vector for incoming call processing at eCommerce location 1.

Primary vector

```
1.   wait time 0 secs hearing ringback
2.   consider split1 pri m    adjust-by 0
3.   consider location 2      adjust-by 30
4.   consider location 3      adjust by 10
5.   queue-to-best
```

For this example, assume that location 2 returned the lowest EWT, so the call will be routed to that site.

Status poll vector

To collect information from the remote switch, the command **`consider location 2 adjust-by 30`** in the primary vector places a status poll using IP trunking to the status poll vector on the switch at location 2. The following example provides an example status poll vector on the remote switch.

Status poll vector

```
1.   consider split2 pri m              adjust-by 0
2.   consider split 11pri m             adjust-by 0
3.   reply-best
```

The status poll only obtains information and returns it to the origin switch; the call is not connected to the status poll VDN. Once the remote switch has returned the necessary information, the **`consider`** series in the primary vector at location 1 can continue at the next vector step.

Interflow Vector

Once the switch has selected the site to which the call should be routed (location 2), the call is sent to the public network. The public network switch then sets up the second leg of the call and passes the codeset 0 UUI information in the SETUP message if this is supported. Next, the Avaya switch tells the public switch to transfer the call over the public network. The Avaya switch knows to do this because Net Redir for location 1, location 2, and location 3 was set to y on the BSR Application Form.

For incoming 800 number calls from MCI WorldCom DMS-250 network switches, the vector reached by the second leg call placed by the switch must immediately be answered (and send an ISDN CONNect message). This can be accomplished with a **`wait 0 secs hearing music`** or an **`announcement`** step as the first step in the receiving interflow vector. The following example shows an example interflow vector for eCommerce location 2.

BSR example of interflow vector on remote switch

```
1.   announcement83345
2.   consider split2 pri m              adjust-by 0
3.   consider split 11pri m             adjust-by 0
4.   queue-to best
```

The public network then merges the second leg of the call to the second site and drops the trunk to the Avaya switch.

BSR using EWT and agent adjustments example

In this example, a catalog company has three contact centers, two in the United States and one in France. BSR is implemented across the sites. The catalog company uses the UCD-MIA call distribution method at each site and uses the UCD-MIA available agent strategy for the VDN that is active for the call. The catalog company will use the **`adjust-by`** option in the **`consider`** vector step to select the best agent at any site to receive a call.

The catalog company uses the **`adjust-by`** command to consider delivery of calls based on adjusted idle times for the agents, so that a remote site is not selected when agent idle time differences are not significant.

To activate the BSR Available Agent Adjustment option, the administrator sets the “BSR Available Agent Adjustments” field on the Feature-Related System Parameters form to **y**.

To use the option, the switch Administrator changes the **`adjust-by`** value in the **`consider`** vector steps to include a percentage adjustment appropriate for each contact center. In this example, adjust-by values are defined as 0 for the first contact center, 20% for the second contact center, and 20% for the third contact center. If there is an agent surplus at two or more of the contact centers, then the adjustment will apply. The adjustment makes sites more or less desirable, based on decreasing the idle time of available agents by the percentage assigned for the site.

Note:

If the actual agent idle time is 100 or more seconds, then the idle time is decreased by the assigned percentage. If the actual agent idle time is less than 100 seconds, then the idle time is decreased by the adjustment in seconds.

The following table summarizes how the above adjustment can affect the idle times for each site.

Idle time adjustment calculations

	Agent idle time	Adjust by xx%	Calculation	Adjusted idle time
incoming split 1 at location 1	40	0[1]	0	40
location 2	50	20	50 - 20 secs	30
location 3	100	20	100 - 20 secs (20% of 100)	80

1. Since the adjust-by value in this consider step is set to zero, no adjustment is made.

Primary Vector

An incoming call arrives at location 1 and is processed by the primary vector. This vector begins the BSR process by considering the specified resources.An example primary vector for incoming call processing at location 1 is shown in the following example.

Primary vector with adjustments

```
1.   wait time 0 secs hearing ringback
2.   consider split1 pri m     adjust-by 0
3.   consider location 2       adjust-by 20
4.   consider location 3       adjust-by 20
5.   queue-to-best
```

In this example, the **consider** commands in steps 2, 3, and 4 collect information to compare local split 1 with location 2 and location 3. In each case, an available agent is found and an agent idle time returned. The **adjust-by** in steps 3 and 4 adjusts the value of the agent idle time as shown in table Idle time adjustment calculations on page 93. Step 5 queues the call to the best location found.

Status poll vector

To collect information from the remote switch, the command **consider location 2 adjust-by 20** in the primary vector places an ISDN call (a status poll) to the status poll vector on the switch at location 2. The example status poll vector is shown below.

Status poll vector

```
1.   consider split2 pri m               adjust-by 0
2.   consider split 11pri m              adjust-by 0
3.   reply-best
```

The status poll only obtains information and returns it to the origin switch; the call is not connected to the status poll VDN.

This vector compares splits 2 and 11, identifies the better of the two, and sends this information back to switch 1 with the **`reply-best`** command. Notice that the **`adjust-by`** command could be used on the remote switch to adjust the EWT or agent idle time that is returned by either of the splits. When adjustments are applied at both the origin and remote switches, the two adjustments are added at the origin switch.

The **`consider`** command is ISDN-neutral and does not return answer supervision. The status poll call is dropped when the **`reply-best`** step executes, but the ISDN DISCONNect message returned to switch 1 contains the information from the best split considered at location 2. Once the remote switch has returned the necessary information, the consider series in the primary vector on switch 1 can continue at the next vector step.

Interflow Vector

Based on the values derived in table Idle time adjustment calculations on page 93, at each site, location 2 is the best site based on the adjusted agent idle time. The **`queue-to best`** command in the primary vector interflows the call to the interflow vector at location 2. The example interflow vector is shown below.

Interflow vector on remote switch

```
1.   consider split2 pri m              adjust-by 0
2.   consider split 11pri m             adjust-by 0
3.   queue-to best
```

The interflow vector reconsiders the status of both splits to get the most current information and queues or delivers the call to the best split. Notice that the **`consider`** sequences in the interflow vector and the status poll vector are identical except for the last step.

When the call is interflowed, it is removed from any queues at the origin switch and any audible feedback at the origin switch is terminated.

Dial by Name

The Dial by Name feature allows you to “dial” someone by entering the person’s name from your touch-tone keypad. This feature is accessible by using the Call Vectoring feature and the integrated announcement circuit pack to create an “auto-attendant” procedure in which one of the options allows callers to enter a person’s name instead of the person’s extension number. The system processes the name characters received, and, when a match is found, the number is dialed automatically.

Note:

The Dial by Name feature must be enabled to create a vector for this purpose.

A typical scenario includes the following call processing features:

- When a call comes in to the system (usually to a Listed Directory Number), a vector routes the call to an announcement that says, “Hello. You have reached A1 Hotel. Please press 0 for the operator; press 1 for the front desk; press 2 if you know the guest’s extension; press 3 if you know the guest’s name; press 4 if you want to choose from a list of extensions; or press 5 if you wish to hear these options again.”
- When the caller selects 3, the caller is then instructed to enter the person’s name.
- As soon as a single match is found, the call is placed to that person.

You can assign several vectors that define how calls will be handled as users select the different prompts. The following example shows an “auto-attendant” procedure that can be used to access the Dial by Name feature. Step numbers 1-20 contain the basic auto-attendant steps, and steps 21-32 contain the Dial by Name steps.

Example Dial by Name vector

```
change vector 2                                              Page   1 of   3
                                   CALL VECTOR

    Number: 2                 Name: Dial by Name
                   Attendant Vectoring? y            Lock? n
     Basic? y   EAS? n    G3V4 Enhanced? n    ANI/II-Digits? n   ASAI
Routing? n
 Prompting? y   LAI? n   G3V4 Adv Route? n    CINFO? n  BSR? n    Holidays? y

01 wait-time    2   secs hearing ringback
02 collect      1  digits after announcement 381
03
04 route-to     number 0                with cov n if digit             =  0
05 route-to     number 105              with cov n if digit             =  1
06 goto         step   12  if digits           =      2
07 goto         step   21  if digits           =      3
08 goto         step   19  if digits           =      4
09 goto         step   16  if digits           =      5
10 route-to     number 0                with cov n if unconditionally
11
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

change vector 2                                         Page 2 of 3
                             CALL VECTOR

12 collect      3  digits after announcement 382
13 route-to     digits with coverage y
14 route-to     number 0               with cov n if unconditionally
15
16 goto         step   2   if unconditionally
17
18
19 collect      3  digits after announcement 383
20 goto         step   13  if unconditionally
21 collect      4 digits after announcement 661
22 route-to     name1 with coverage y
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

change vector 2                                         Page 3 of 3
                             CALL VECTOR

23 goto         step 30 if nomatch
24 collect      11 digits after announcement 662
25 route-to     name2 with coverage y
26 goto         step 30 if nomatch
27 collect      2 digits after announcement 663
28 route-to     name3 with coverage y
29 goto         step 30 if nomatch
30 collect      1 digits after announcement 660
31 goto         step 21 if digits = 1
32 route-to     number 0                with cov n if unconditionally
```

This example includes the following call processing features and functionalities:

1. When someone calls the system, the caller receives ringback for 2 seconds.
2. Announcement 381 plays. This announcement asks the caller to do one of the following:
   - Press 0 if the caller wants the operator; if the caller presses 0 or waits for the timeout, the call is routed to the operator.
   - Press 1 if the caller wants the front desk; if the caller presses 1, the call is routed to extension 105, which is the front desk.
   - Press 1 if the caller knows the person's extension; if the caller presses 2, the call is routed to announcement 382, which instructs the caller to dial the person's extension.
   - Press 3 if the caller knows the person's name; if the caller presses 3, the following sub-procedure occurs:
     i. Announcement 661 plays requesting that the caller enter the first four characters of the person's last name.
        — If there is a single match, the call is redirected.
        — If there are multiple matches, continue with ii.
        — If there is no match, go to iv.
     ii. Announcement 662 plays requesting that the caller enter the rest of the person's last name, followed by the # key.
        — If there is a single match, the call is redirected.
        — If there are multiple matches, continue with iii.
        — If there is no match, go to iv.
     iii. Announcement 663 plays requesting that the caller enter the first two characters of the person's first name.
        — If there is a single match, the call is redirected.
        — If there is no match, continue with iv.
     iv. Since there are still no matches, announcement 660 plays telling the caller that he or she can press 1 to try again, or press 0 to get an operator.
   - Press 4 if the caller knows the department (such as housekeeping) that he or she wishes to access; if the caller presses 4, the call is routed to announcement 383, which gives the caller a list of several departments that the caller can dial directly.
   - Press 5 to start over again; if the caller presses 5, the caller hears announcement 381, which repeats all of the options.
   - If the caller dials anything else, the call is routed to the operator.

Vectors exercises

This section presents several typical business scenarios that involve telephone use. One or more vectors are provided that show how to handle each of these scenarios.

The vectors presented here are intended to be “suggested solutions.” Individual contact centers must consider their own unique requirements and budget in selecting and writing vectors.

Emergency and routine service

Write a vector that does the following:

- Delivers the following message to handle emergency calls: “We are aware of the power outage in the northeastern part of the city. Crews have been dispatched. If you are calling for other reasons, please hold for an operator.”
- Enables the caller to speak with an agent, if an agent is available, concerning a nonemergency matter.

Suggested solution 1

Call Vectoring option

```
1.   wait-time 0 seconds hearing ringback
2.   announcement 4100 ("We are aware of the
     power outage in the northeastern part of the city. Crews have
     been dispatched.  If you are calling for other reasons, please
     hold for an operator.")
3.   wait-time 2 seconds hearing ringback
4.   goto step 10 if calls-queued in split 1 pri l > 20
5.   queue-to split 1 pri l
6.   wait-time 6 seconds hearing music
7.   announcement 4200 ("We’re sorry.  All of
     our operators are busy.  Please hold.")
8.   wait-time 10 seconds hearing music
9.   goto step 7 if unconditionally
10.  disconnect after announcement 4200 ("We’re sorry.
     All of our operators are busy at the moment. Please call back at
     your convenience.")
```

In step 2 of the example vector shown above, the **`announcement`** command provides the caller with the appropriate emergency information, and it invites the caller to hold if he or she wants to speak with an operator on another matter. If the caller holds, the caller hears several seconds of ringback provided by the **`wait-time`** command in step 3. Thereafter, the **`goto step`** command in step 4 checks whether there are more than 20 calls queued in split 1. If so, a branch is made to step 10, where the **`disconnect after announcement`** command first informs the caller that the call cannot be serviced at this time and then drops the call.

On the other hand, if 20 or fewer calls are queued to split 1, the call is queued to the split by the **queue-to split** command in step 5. Thereafter, unless the call is answered, feedback in the form of music is provided by step 6 and an announcement urging the caller to hold is provided by step 7. After another wait with music period (if necessary) that is provided by step 8, the **goto step** command in step 9 branches back to the aforementioned “please hold” announcement in step 7. The resulting “announcement-wait” loop (steps 7 through 9) is then repeated until either an agent answers the call or the caller hangs up.

Suggested solution 2

Note:

This example uses the Call Prompting feature. For more information about Call Prompting, see Chapter 10: Call Prompting on page 181.

Call Vectoring and Call Prompting option

```
VDN (extension=1030   name="Hub"   vector=30)
Vector 30:
      1. wait-time 0 seconds hearing ringback
      2. collect 1 digits after announcement 3000
         ("We are aware of the power outage in the northeastern
         part of the city.  Crews have been dispatched.  If
         you are calling for other reasons, please press 1.
         Otherwise, please hang up now.")
      3. route-to number 1031 with cov y if digit = 1
      4. announcement 3100 ("Entry not understood.  Please
         try again.")
      5. goto step 2 if unconditionally

VDN (extension=1031   name="Service"   vector=31)
Vector 31:
      1. announcement 4000 ("Please hold.  We will
         try to connect you to an operator.")
      2. wait-time 2 seconds hearing ringback
      3. goto step 9 if calls-queued in split 1 pri l > 20
      4. queue-to split 1 pri l
      5. wait-time 6 seconds hearing music
      6. announcement 4200 ("We're sorry.  All of
         our operators are busy.  Please hold.")
      7. wait-time 10 seconds hearing music
      8. goto step 6 if unconditionally
      9. disconnect after announcement 4200 ("We're
         sorry.  All of our operators are busy at the moment.
         Please call back at your convenience.")
```

Suggested Solution 2 involves both Call Vectoring and Call Prompting. Also, it involves two vectors instead of just one vector, and it assumes the that caller is calling from a touchtone telephone. The announcement portion of the **collect digits after announcement** command in step 2 of Vector 30 first provides the caller with the appropriate emergency information. It then invites the caller to press “1” if the caller is calling for some other reason. If this is not the case, it finally suggests that the caller hang up.

Assume that the caller wants to hold the line but enters the incorrect touchtone digit (**2**, for example). In such a case, the **route-to number** command in step 3 attempts to route the call to VDN extension 1031 according to the entered digit. However, because a number other than **1** was entered, the call is not routed to the VDN extension. Instead, control is passed to step 4, where the **announcement** command first informs the caller of the input error and then invites the caller to try again. Thereafter, the **goto step** command in step 5 unconditionally sends control back to step 2, where the **collect digits** command ultimately collects the digit that was entered by the caller. The digit-input loop (steps 2 through 5) continues for as long as the caller enters an incorrect digit.

If the caller correctly enters digit **1** as requested by the **collect digits** command in step 2, the **route-to number** command in step 3 sends control to the vector whose VDN extension is 1031, (Vector 31).

Late Caller Treatment

The contact center is staffed by union agents who work under a contract that stipulates that agents are free to leave promptly at 5:00 p.m. However, you are concerned about the callers who will call shortly before 5:00 p.m. on any given day and find themselves waiting in queue the at the top of the hour.

Write a vector that warns late callers that their call may not be serviced. Remember that business hours are from 8:00 a.m. to 5:00 p.m., Monday through Friday.

Suggested solution:

Late caller treatment

```
1.   goto step 15 if time-of-day is all 1700 to all 0800
2.   goto step 15 if time-of-day is fri 1700 to mon 0800
3.   goto step 16 if calls-queued in split 1 pri l > 20
4.   queue-to split 1 pri l
5.   goto step 10 if time-of-day is all 1645 to all 1700
6.   wait-time 20 seconds hearing ringback
7.   announcement 100 ("We're sorry, all of our
     agents are busy...Please hold...")
8.   wait-time 998 seconds hearing music
9.   stop
10.  announcement 200 ("It is almost closing time.
     We will try to service you before we close for the day.
     However, if we are unable to do so, please call back
     at your convenience between 8:00 A.M. and 5:00 P.M.,
     Monday through Friday.")
11.  wait-time 30 seconds hearing music
12.  goto step 14 if time-of-day all 1700 to all 1710
13.  goto step 11 if unconditionally
14.  disconnect after announcement 300 ("We're sorry, our office is now closed.
     Please call back at your convenience between 8:00 A.M.     and 5:00 P.M.,
     Monday through Friday.")
15.  disconnect after announcement 400 ("We're sorry, our office is     closed.
     Please call back at your convenience between 8:00 A.M.     and 5:00 P.M.,
     Monday through Friday.")
16.  disconnect after announcement 500 ("We're sorry, we cannot service your
     call at this time. Please call back at your convenience between
     8:00 A.M. and 5:00 P.M., Monday through Friday.")
```

In the example vector shown above, specific treatment is provided for calls that come into the switch after working hours, during the weekend, or as the working day comes to a close.

The **`goto step`** command in step 1 checks whether the call is placed during nonworking hours during the week. If the call is received at this time, a branch is made to step 15, where the **`disconnect after announcement`** command first informs the caller that the office is closed and then drops the call. If the call is not received at the time specified in step 1, control is passed to step 2, where another **`goto step`** command checks whether the call is received during weekend hours. If the call is received during weekend hours, a branch is made to step 15. If the call is not being placed at this time, control is passed to step 3.

The **`goto step`** command in step 3 checks for the number of calls in split 1. If more than 20 calls are queued to split 1, control is passed to step 16, where the **`disconnect after announcement`** command first informs the caller that the call cannot be serviced at this time and then disconnects the call. If 20 or fewer calls are queued to split 1, control is passed to step 4, where the **`queue-to split command`** queues the call to split 1.

Control is then passed to step 5, where the **goto step** command checks whether the current time is any time between 4:45 p.m. and 5:00 p.m. inclusive (very close to, if not, closing time). If the current time does not fall within this clock range, the **wait-time** command in step 6 provides the caller with 20 seconds of ringback. Thereafter, the **announcement** command in step 7 plays the appropriate hold message, and the **wait** command in step 8 provides the caller with 998 seconds of music. Finally, the **stop** command in step 9 halts vector processing, and the call remains in queue until either the agent answers the call or the caller hangs up.

If the current time is 4:45 p.m. to 5:00 p.m. step 5 executes a branch to step 10, where the appropriate late caller announcement is provided to the caller. Thereafter, the **wait-time** command in step 11 provides the caller with 30 seconds of music. Control is then passed to step 12, where the **goto step** command checks whether the time is currently any time between 5:00 p.m. and 5:10 p.m., inclusive. If so, control is passed to step 14, where the **disconnect after announcement** command first informs the caller that the office is now closed and then invites the caller to call back at the appropriate time before finally disconnecting the call.

If the time is currently not between 5:00 p.m. and 5:10 p.m,. inclusive, control is passed to step 13, where the **goto step** command branches back to the **wait-time** command in step 11. The resulting loop consisting of steps 11 through 13 is repeated for as long as the time is between 5:00 p.m. and 5:10 p.m., inclusive, or until the caller hangs up. Once step 12 is executed at least a second after 5:10 P.M., control is passed to step 14 as described previously.

Messaging option

Write a vector that:

- Does the following if the oldest call waiting is in queue for longer than 75 seconds:
  - — Sends the call to AUDIX (if possible)
  - — Delivers to the caller the following personalized AUDIX message: “All of our MegaSports agents are busy...Please leave your name and telephone number.”
- Plays 30 seconds of ringback for the caller
- After the ringback, plays an announcement for the caller that is followed by music

Suggested solution:

Messaging option

```
 1.  goto step 8 if oldest-call-wait in split 50 pri l > 74
 2.  goto step 8 if calls-queued in split 50 pri l > 20
 3.  queue-to split 50 pri l
 4.  wait-time 30 seconds hearing ringback
 5.  announcement 1000 ("All of our MegaSports
     agents are busy...Please wait...")
 6.  wait-time 998 seconds hearing music
 7.  stop
 8.  announcement 2000 ("We’re sorry, all of our
     MegaSports agents are busy.  If you’d like to leave a
     message, please do so after the tone.  Otherwise, please
     call back between 8:00 A.M. and 5:00 P.M, Monday through
     Friday. Thank you.")
 9.  messaging split 20 for extension 4000
10.  disconnect after announcement 2050 ("We’re sorry, we are unable
     to take your message at this time.  Please call back
     between 8:00 A.M. and 5:00 P.M., Monday through Friday.
     Thank you.")
```

The **`goto step`** command in step 1 of the example shown above checks whether the oldest call waiting in split 50 has been waiting for 75 seconds or more. If so, control is passed to step 8, where the **`announcement`** command first informs the caller that all of the agents are busy and then invites the caller to either call back at the appropriate time or leave a recorded message for the agent. If the caller chooses to leave a message, the **`messaging split`** command in step 9 is executed. Upon execution of the **`messaging split`** command, an attempt is made to connect the caller to AUDIX so that he or she can leave a recorded message. If the split queue is full, or if the AUDIX link is out of service, termination to AUDIX is unsuccessful, and vector processing continues at the next vector step. This step, as is the case here, usually contains an announcement that provides the caller with the appropriate apology and subsequent directives. If the caller is successfully connected to AUDIX, vector processing terminates, and a message can be left for the specified mailbox (4000, in this case).

In step 1, if the oldest call waiting in split 50 has been waiting for fewer than 75 seconds, control is passed to step 2, where another **`goto step`** command checks for the number of calls in split 50. If more than 20 calls are queued to split 50, control is passed to step 8. Thereafter, the procedure for the messaging option that is provided in the previous paragraph is implemented. If there are 20 or fewer calls waiting in split 50, control is passed to step 3, where the **`queue-to split`** command queues the call to the split.

Chapter 5: Basic Call Vectoring

Basic Call Vectoring allows you to program the type of call treatment that a telephone call receives. You program using a set of vector commands. The vector commands that are available to you as part of the Basic Call Vectoring feature set are the simplest and most common commands that are used to program call vectors.

Vector commands can direct calls to various destinations, such as splits, adjuncts or other vectors. The commands can also direct calls to various treatments, such as announcements, a forced disconnect, a forced busy, or a delay treatment.

Basic Call Vectoring includes the following topics:

Command set

The following table shows the commands used for Basic Call Vectoring.

Command category	Action taken	Command
Treatment		
	Play an announcement.	**announcement**
	Delay with audible feedback of silence, ringback, system music, or alternate audio or music source.	**wait-time**
	Play a busy tone and stop vector processing.	**busy**
	Disconnect the call.	**disconnect**
	Execute a Voice Response Unit (VRU) script.	**converse-on split**
Routing		
	Queue the call to an ACD split.	**queue-to split**
	Queue the call to a backup ACD split.	**check split**
	Leave a message.	**messaging split**
	Route the call to a number that is programmed in the vector or to a Service Observing Feature Access Code.	**route-to number**
	Send a message to an adjunct that requests routing instructions for the call.	**adjunct routing**
Branching/Programming		
	Go to a vector step.	**goto step**
	Go to another vector.	**goto vector**
	Stop vector processing.	**stop**

Basic Call Vectoring allows you to use vectoring commands from each of the above Command Categories to process telephone calls. The following sections explain the commands in more detail.

Treatment commands

Call treatment is the type of feedback the caller receives if the caller is not immediately connected to an agent. Basic Call Vectoring includes the ability to implement several types of call treatment commands.

announcement command

The **`announcement`** command connects calls to a recorded announcement.

Announcements can be classified into three groups:

- Delay announcements
- Forced announcements
- Information announcements

Depending on the type of announcement equipment and how the equipment is administered, callers may be required to listen to an entire announcement or they may be able to interrupt an announcement as it is playing.

When a call is connected to an announcement, any previous treatment is discontinued.

For announcements that always start at the beginning, the caller may have to wait in an announcement queue if the announcement is not ready to play. Callers hear the previously established call treatment (if any) until the announcement starts. If the announcement queue is full, vector processing retries the **`announcement`** command indefinitely.

Note:

If an integrated announcement board is in use and the requested announcement is not administered or recorded, vector processing skips the **`announcement`** command and continues with the next vector command.

If the call is in a split/skill queue, the call remains in queue while the announcement plays. If the call is still in queue after the announcement ends, the caller hears silence until another **`announcement`** command, a **`wait hearing ringback`** command, or a **`wait hearing music`** command is processed. If the call connects to a station while the announcement is playing, the announcement stops and the caller hears ringback.

When the announcement completes and is disconnected, the caller hears silence until either a vector step with alternate treatment is processed or the call reaches an agent's station.

Delay announcements

An example of a delay announcement is shown in the following figure.

Delay announcement example

```
announcement 2556 (‘‘All our agents are busy.
     Please hold.’’)
```

If the caller remains on hold, a supplementary delay announcement similar to the following example can be used.

Supplementary delay announcement example

```
announcement 2557 (‘‘Thanks for holding.  All
     our agents are still busy.  Please hold.’’)
```

A delay announcement is usually coupled with a delay step, which is provided by the **wait-time** command. For more information about the **wait-time** command, see wait-time command on page 109.

Forced announcements

In some circumstances, a contact center may not wish to allocate resources to certain calls. Usually, this option is exercised when heavy call traffic is expected due to a major event such as a widespread service problem which is currently being addressed. An appropriate announcement can be inserted into a vector to address such circumstances. An example of such a forced announcement is shown below.

Forced announcement example

```
announcement 1050 (‘‘We are aware of the current
    situation and are working to rectify the problem.  If your
    call is not urgent, please call back later.’’)
```

Information announcements

In some circumstances, a contact center can anticipate that the caller can be provided with recorded information that fully addresses their needs, so that no further interaction is required. An example of such an announcement, which is referred to as an information announcement is shown in the following example.

Information announcement example

```
disconnect after announcement 2918 (''Today has
     been declared a snow day.  Please report for work tomorrow
     at 8 A.M.'')
```

A forced announcement is followed by a **`disconnect`** command is used with the announcement. After the announcement, the caller is disconnected, since he or she need not stay on the line any longer.

wait-time command

The **`wait-time`** command enables you to create a vector that delays the call with audible feedback. In presenting an example of a delay announcement earlier in this chapter, we mentioned that this type of announcement is usually coupled with a delay step. A delay step is provided by the **`wait-time`** command, which allows the caller to remain on hold for at least the amount of time that is indicated in the command.

The following example shows an announcement that includes the **`wait-time`** command in a delay step.

Call delay with audible feedback

```
announcement 2556 (''All of our agents are busy.
     Please hold.'')
     wait-time 20 seconds hearing music
```

In the example shown above, the caller waits at least 20 seconds for the call to be answered by an agent. During this wait period, the caller is provided with system music, which is one type of feedback that is available with the **`wait-time`** command.

If the delay step is the final effective step in the vector, the audible feedback continues beyond the specified duration. In a vector, a final effective step is defined as the last vector step, or a vector step that is followed by a **`stop`** step.

Under normal circumstances, the audible feedback continues until the call is either answered or abandoned. However, if the call is not queued when vector processing stops, the call is dropped. Feedback also continues while a call is queued to a converse split, that is, any split routed to by a **`converse-on split`** command, and while data is being passed to a Voice Response Unit (VRU). Finally, feedback also continues during the wait period before the connection of an announcement and/or a Touch-Tone Receiver (TTR).

For more information about TTRs, which are used with the Call Prompting feature, see Chapter 10: Call Prompting on page 181.

Multiple audio or music sources on delay

You can specify an alternative audio or music source for a vector **wait-time** step. This alternative source can be any extension number that is administered on the Announcements/Audio Sources form. For instructions for entering an audio or music source on this form, see *Administrator Guide for Avaya MultiVantage Software*, 555-233-506.

With the Multiple Audio/Music Sources feature, you can tailor the **wait-time** feedback to the interests, tastes, or requirements of the audience. You can provide specific types of music or music with overlays of advertising that relate to the service provided by the splits or skills that the vector serves. Or, additional advertising messages can be heard by the callers as they wait for an available agent.

An example of an announcement that includes an alternative audio or music source in the **wait-time** step is shown below.

Call delay with multiple audio/music source feedback

```
announcement 2556 (‘‘All of our agents are busy.
     Please hold.’’)
     wait-time 20 seconds hearing 55558 then music
```

When the **wait-time** step is encountered, the caller is connected to extension 55558 for 20 seconds. At the end of 20 seconds, the next vector step is executed. The “then” option in the **wait-time** step specifies what the caller hears if the caller cannot be connected to the specified source. Or, when the call is waiting in queue, the “then” option specifies what the caller hears if the call is not answered in 20 seconds. In this example, if the call is not answered in 20 seconds, the caller hears system music until a subsequent **announcement**, **busy**, **collect**, **converse-on**, **disconnect** or **wait-time** step is encountered.

You can specify **music** (system music), **ringback**, **silence**, or **continue** for the **then** option. When continue is specified, the caller continues to hear the alternative audio or music source until it is replaced by a subsequent vector step regardless of the time specified in the **wait-time** step.

You can use alternate audio or music sources in vector loops to provide continuous audible feedback as shown in the following example vector steps.

Call delay with continuous audible feedback

```
1. ...
2. ...
3. ...
4. wait-time 30 secs hearing 55558 then continue
5. route-to number 913034532212 with cov n
6. goto step 4 if unconditionally
```

In the example shown above, a look-ahead call attempt is placed every 30 seconds on behalf of the caller. If extension 55558 is a long, barge-in, repeating announcement, the caller hears announcement 55558 all the way to the end without the announcement being restarted each time vector processing returns to step 4.

Multiple music sources on hold

This feature uses the tenant partitioning tenant number (TN) to determine which music source to use when a call is put on hold. You can assign a different music source to each possible TN.

For more information, about tenant partitioning, see *Administrator Guide for Avaya MultiVantage Software*, 555-233-506.

Without EAS, the COR setting of the station or extension that puts the call on hold determines whether music-on-hold is applied. With EAS, the COR setting of the logical agent ID is used to determine whether music-on-hold is applied. The TN that is assigned to the destination extension number determines the music source. You assign a music source number to the TN on the Tenant form. The physical location (port) of the music source is assigned on the Music Sources form.

The TN that is assigned to the active VDN on the VDN form determines the music source that is used. During vectoring, a **`wait hearing music`** command attaches the vector delay music source that is defined by the TN for the active VDN. The Multiple Music Sources for Vector Delay, which is a **`wait hearing extension then...`** command, applies the vector delay source that is defined by the **Announcements** or **Audio Sources** extension regardless of the TN that is assigned to the VDN.

Note:

> The TN administered for extensions on the **Announcement or Audio Sources** form applies only to direct calls to the announcement extension. For these calls, the announcement or music source assigned to the TN is what the caller hears.

During vector processing, if the **`converse`** vector command connects the call to an agent when the call remains under vector control and the agent puts the call on hold, the active VDN applies music-on-hold.

When a vector routes a call to another destination by a **`queue`**, **`check`**, **`route-to`**, or **`messaging split`** command, the switch uses the TN of the last active VDN to determine the music source for music-on-hold.

In ACD systems without vectoring and where music-on-hold applies, the TN assigned to the called hunt group extension determines which music source callers hear while in queue or on hold.

busy command

A busy tone and subsequent termination of vector processing are produced using the **busy** command. An exception to this occurs on CO trunks where answer supervision has not been sent. Callers on such trunks do not hear the busy tone from the switch. Instead, these callers continue to hear ringback from the CO. The **busy** command eventually times out and drops the call after 45 seconds. With ISDN PRI, busy tone can be provided from the network switch.

You might want to force a busy tone to process a call that arrives at a time when there are a large number of calls queued in the main split, or when the contact center is out of service or closed.

An example vector that demonstrates the **busy** command is shown below.

Busy command example

```
1. goto step 6 if calls-queued in split 1 pri h > 30
2. queue-to split 1 pri h
3. announcement 4000
4. wait-time 2 seconds hearing music
5. stop
6. busy
```

In the example vector shown above, the **goto step** command in step 1 sends call control to **busy** in step 6 if the conditions in the former command are met. Specifically, if the number of calls that are queued at a high priority is greater than 30, the **busy** command is accessed.

disconnect command

The **disconnect** command forcibly disconnects a call with an optional announcement. Any previously established call treatment ends when the **disconnect** command is executed, and the call is removed from vector processing and from the switch.

If the call is connected to a station while the announcement is playing, the announcement stops and the caller hears ringback. Also, because vector processing stops when the call connects to a station, the disconnect portion of the command is not processed.

When the **disconnect** command includes an announcement, the switch sends answer supervision (if it was not already sent) just before the announcement plays.

When the **disconnect** command does not include an announcement, the switch sends answer supervision before it disconnects a call.

Note:

Answer supervision is not sent for ISDN trunks.

An example of the **disconnect** command is shown below.

Call disconnect example

```
disconnect after announcement 2918 (‘‘Today has
     been declared a snow day.  Please report for work tomorrow
     at 8 P.M.’’)
```

In this example, the caller is provided with sufficient information to meet their needs, so that no further interaction is required.

converse-on split command

Voice Response Integration (VRI) is designed to enhance the integration of the Call Vectoring with the capabilities of voice response units (VRUs), particularly the Conversant Voice Information System.

VRI can do the following:

- Execute a VRU script while retaining control of the call in the vector processing.
- Execute a VRU script while the call remains in the split queue and retains its position in the queue.
- Group Conversant ports for multiple applications. This ability was previously possible only when ASAI was in use.
- Use a VRU as a flexible external announcement device.
- Pass data between the switch and a VRU.
- Tandem VRU data through the switch to an ASAI host.

The capabilities in the previous list are provided by the **`converse-on split`** command, which is an enhancement to the Basic Call Vectoring customer option. The **`converse-on split`** step is specifically designed to integrate a VRU with the DEFINITY. VRI allows VRU capabilities to be used while keeping control of the call in the DEFINITY. The inclusion of VRUs with vector processing provides the following advantages:

- Access to local and host databases
- Validation of caller information
- Text to speech capabilities
- Speech recognition
- Increased recorded announcement capacity
- Audiotex applications
- Interactive Voice Response (IVR) applications
- Transaction processing applications

One of the advantages of VRI is that it allows users to make more productive use of queuing time. For example, while the call is waiting in queue, the caller can listen to product information by using an audiotex application or by completing an interactive voice response transaction. In some cases, it may even be possible to resolve the caller's questions while the call is in queue. This can help reduce the queuing time for all other callers during peak intervals.

In addition, when Advanced Vector Routing is enabled, the Expected Wait Time for a call can be passed to the VRU. In this way, the caller can be told how much longer she can expect to wait before her call will be answered. See Expected Wait Time (EWT) on page 126 for a complete description of the EWT feature.

During the execution of a VRU script, if the caller previously queued to an ACD split, the caller retains his or her position in queue. If an agent on the switch becomes available to service the call, the line to the VRU is immediately dropped, and the calling party is connected to the available agent.

An example of a vector that can access voice response scripts from a VRU is shown below. This example also shows one way in which more than one VDN can access the same vector.

Accessing voice response scripts

```
VDN (extension=1040   name=''car loans''     vector=40)
VDN (extension=1041   name=''equity loans''  vector=40)
Vector 40
     1. goto step 10 if calls-queued in split 1 pri h > 30
     2. queue-to split 1 pri h
     3. announcement 4000
     4. goto step 7 if calls-queued in split 1 pri h < 5
     5. wait-time 0 seconds hearing music
     6. converse-on split 11 pri h passing vdn and none
     7. wait-time 20 seconds hearing music
     8. announcement 4001
     9. goto step 7 if unconditionally
    10. busy
```

For this example, assume that a caller wants to hear information that concerns car loans. Also assume that the call is queued to split 1 (step 2) and that vector processing proceeds to step 6. In this case, the **`converse-on split`** command in this step delivers the call to the converse split if there is a queue for the split and the queue is not full, or if a VRU port is available. Otherwise, vector processing continues at the next vector step. When the VRU port responds, the step then outpulses VDN 1040 to the VRU by way of the **`passing vdn`** subcommand that is included in the command. In turn, the VRU executes the "car loans" voice response script for the caller. Note that it is important to provide a feedback step prior to the converse-on step in case there is a delay in reaching an available converse split port. In this example, step 5 provides music for this purpose.

Now, assume that another caller wants information that concerns equity loans. In this case, VDN 1041 is outpulsed to the VRU, which in turn executes the "equity loan" voice response script for the caller.

In either case, while interaction with the VRU is taking place, the call remains in the appropriate split's queue (split 1 in this example). If an agent answers the call while the voice response script is being executed, the voice response script is interrupted, the line to the VRU is dropped, and the caller is connected to the available agent. Once a voice response script starts, no further vector steps are executed until the voice response script is complete.

For more information about the call flow for converse-VRI calls, see Appendix J: Call flow and specifications for converse – VRI calls on page 585.

Besides VDN extensions, the **`converse-on split`** command can outpulse to the VRU calling party extensions, collected (inputted) caller digits (if Call Prompting is enabled), Expected Wait Time (if Advanced Vector Routing is enabled) call queue positions, a string of a maximum of six digits or asterisks, a pound sign (#), or nothing. For more information, see Advanced Vector Routing - EWT and ASA on page 125, Chapter 10: Call Prompting on page 181, and in Appendix A: Call Vectoring commands on page 387.

Note:

In vector example Accessing voice response scripts on page 114, the **`calls-queued`** condition in the second **`goto`** step (step 4) in effect serves as a checkpoint for determining whether there is enough time for the voice response script, which is activated by the **`converse-on`** step, to be executed. Specifically, if five or more calls are queued to split 1, it is considered feasible to execute the voice response script.

Routing commands

Basic Call Vectoring includes vectoring commands that enable you to route telephone calls.

Note:

Adjunct routing is fully described in Adjunct (ASAI) Routing on page 163.

queue-to split and check split commands

Calls that come in to the Call Vectoring system can be queued to a maximum of three ACD splits. Two commands are used to queue calls to splits.

The **`queue-to split`** command queues a call unconditionally. The command sends a call to a split and assigns a queuing priority level to the call in case all agents are busy.

The **`check split`** command conditionally checks the status of a split for possible termination of the call to that split. The command either connects the call to an agent in the split or puts the call into the split's queue at the specified priority level if the condition specified as part of the command is met.

Multiple split queuing

The term *multiple split queuing* refers to the queuing of a call to more than one split at the same time. The following example vector shows this process.

Multiple split queuing example

```
1. goto step 4 if calls-queued in split 1 pri l >= 10
2. queue-to split 1 pri t
3. wait-time 12 seconds hearing ringback
4. check split 2 pri m if calls-queued < 5
5. check split 3 pri m if calls-queued < 5
6. announcement 3001
7. wait-time 50 secs hearing music
8. goto step 4 if unconditionally
```

To avoid completing vector processing without queuing the call to a split, it is always good practice to check a split's queue before queuing to that split. If the queue is full, alternate treatment such as queuing to an alternate split should be provided. In this vector, if the main split's queue (which has 10 queue slots) is full, the **`goto step`** command in step 1 skips the main split and goes directly to step 4 to check the backup splits. Although calls are queued in step 2 at a top priority, a low priority is specified in step 1 so that calls in queue at all priority levels are counted. If there are 10 or fewer calls in the main split, control is passed to step 2, where the **`queue-to split`** command queues the call to split 1. Once the call is queued, vector processing continues at the next step.

Step 4 contains a **`check split`** command. If the call is not answered by the time step 4 is reached, the **`check split`** in the step attempts to queue the call to a second split. Specifically, the command first determines whether there are fewer than five calls that are queued to split 2. If so, the command then attempts to connect the call to an agent in the split. If such a connection cannot be made, the command puts the call into the split's queue at the specified priority level. Vector processing then continues at the next step. If there are five or more calls queued to split 2, the command fails and vector processing continues at step 5.

Step 5 contains another **`check split`** command and, accordingly, the process described in the previous paragraph is repeated, with one difference: the queuing attempt is made to split 3 instead of to split 2.

Except for the condition check, the circumstances under which the **`check split`** command cannot queue a call are identical to those for the **`queue-to split`** command.

Finally, note that whenever a call is queued to a backup split, the call remains queued to the main split and/or to another backup split if it was already queued to either or both of these splits. Once the call is answered in a split to which it is queued, the call is automatically removed from all the other splits to which it is also queued.

Note:

The **`check split`**, **`queue-to-split`**, and converse-on commands can access only those splits that are vector-controlled. A split is considered to be vector-controlled if **`yes`** is entered in the Vector field of the Hunt Group form. With EAS, Multiple Split Queuing is referred to as Multiple Skill Queuing.

Option with the VDN as the coverage point

The Vector Directory Number (VDN) can be used as the last point in a coverage path. This capability allows the call to first go to coverage and then to be processed by Call Vectoring and/or Call Prompting. The capability also allows you to assign AUDIX to a vector-controlled hunt group and to therefore enable access to these servers using a **`queue-to split`** or **`check split`** command. The result of all this is that call handling flexibility is enhanced.

The following example shows a vector, for which the VDN serves as a final coverage point, that allows the caller to leave a recorded message.

Leaving recorded messages (VDN as the coverage point option)

```
VDN 1 (used in a coverage path)
Vector 1
    1. goto step 7 if time-of-day is mon 8:01 to fri 17:00
    2. goto step 13 if staffed-agents in split 10 < 1
    3. queue-to split 10 pri 1 (AUDIX split)
    4. wait-time 20 seconds hearing ringback
    5. announcement 1000 (‘‘Please wait for voice
       mail to take your message.’’)
    6. goto step 4 if unconditionally
    7. goto step 2 if staffed-agents in split 20 < 1
    8. queue-to split 20 pri 1 (audix split)
    9. wait-time 12 seconds hearing ringback
   10. announcement 1005 (‘‘Please wait for an attendant
       to take your message.’’)
   11. wait-time 50 seconds hearing music
   12. goto step 10 if unconditionally
   13. disconnect after announcement 1008 (‘‘We cannot
       take a message at this time.  Please call back tomorrow.’’)
```

In steps 3 and 8 of the vector example shown above, the caller is given the option of leaving a recorded message, but the **`queue-to split`** command instead of the **`messaging split`** command is used in each case. Thus, the call is actually queued to the AUDIX split.

However, a **`messaging split`** command does not queue the call to the split. Instead, if it is successful, it connects the caller to the split so the caller can leave a message for the specified extension. However, termination to the split may turn out to be unsuccessful due to a factor that cannot be checked by vector processing. For example, the AUDIX link might not be functioning, or all AUDIX ports might be out of service.

As a result of the queuing process, a wait-announcement loop can be included after each **`queue-to split`** step, and the appropriate loop can then be executed until the call is actually terminated to either an AUDIX voice port or to an available message service agent. In this vector, steps 4 through 6 comprise the first wait-announcement loop, and steps 10 through 12 comprise the second such loop.

messaging split command

Basic Call Vectoring allows the caller to leave a message for the customer if the agents at the customer site are not available to take telephone calls. This is done with the help of the **messaging split** command. The following example illustrates use of the **messaging split** command.

Leaving recorded message

```
1. goto step 8 if time-of-day is all 16:30 to all 7:30
2. goto step 10 if calls-queued in split 47 pri l >= 20
3. queue-to split 47 pri m
4. wait-time 12 secs hearing ringback
5. announcement 4001
6. wait-time 60 secs hearing music
7. goto step 5 if unconditionally
8. announcement 4111(‘‘We’re sorry, our office
   is closed.  If you’d like to leave a message, please
   do so after the tone.  Otherwise, please call back
   weekdays between 7:30 A.M. and 4:30 P.M.  Thank you.’’)
9. goto step 11 if unconditionally
10.announcement 4222 (“We’re sorry, all of our agents are busy, please leave
   a message after the tone and we will return your call.”)
11. messaging split 18 for extension 2000
12. disconnect after announcement 4333 (‘‘We’re sorry, we are
    unable to take your message at this time.  Please call back at your
    convenience weekdays between 7:30 A.M. and 4:30 P.M. Thank you.’’)
13. busy
```

In this vector, the **goto step** command in step 1 checks to see if the office is open, and branches to step 8 if the office is closed. This is done to accommodate calls that are made during nonworking hours, when there are no agents available to take telephone calls. Accordingly, step 8 provides the caller with an appropriate announcement and an opportunity to leave a recorded message.

Step 2 checks to see if split 47’s queue (which has 20 queue slots) is full, and branches to step 10 if it is. Steps 3 to 7 queue the call to split 47 and then give audible feedback to the caller.

If the caller chooses to leave a message, the **messaging split** command in step 11 is executed. Split 18 in the command is the AUDIX split. Extension 2000 is the mailbox for split 47 (from step 2).

Upon execution of the **messaging split** command, an attempt is made to connect the caller to AUDIX so that he or she can leave a recorded message. If the split queue is full, or if the AUDIX link is not functioning, termination to AUDIX is unsuccessful, and vector processing continues at the next vector step. As is the case here, this step usually contains an announcement that provides the caller with the appropriate apology and subsequent directives. If the caller is successfully connected to AUDIX, vector processing terminates, and a message can be left for the specified mailbox. In this case, the mailbox is 2000.

Finally, if the supervisor or a group of agents has an Automatic Message Waiting (AMW) lamp for the mailbox used, and if the lamp lights, the relevant party, upon returning, knows that a caller has left an AUDIX message.

route-to number command

The **`route-to number`** command can be used to route calls to a vector-programmed number.

Interflow routing

Calls can be queued to a maximum of three splits. Calls can also be routed to a programmed number in the vector using a process that is known as interflow.

Interflow allows calls that are directed or redirected to one split to be redirected to an internal or an external destination. For Basic Call Vectoring, this destination is represented by a number programmed in the vector. The number is always included in the **`route-to number`** command and it may represent any of the following destinations:

- Attendant or attendant queue
- Local extension
- Remote (UDP) extension
- External number
- VDN

When the **`route-to number`** command is used to chain multiple vectors together to enhance processing capabilities, the following events occur:

1. Vector processing continues at the first step in the vector assigned to the routed-to VDN.
2. The call (if queued) is dequeued.
3. Wait treatment (if any) is disabled.
4. Processing then continues in the receiving vector at step 1.

An example of vectors that demonstrate call interflow is shown below.

Call interflow example

```
VDN (extension=1000   name=''Billing Service''  vector=55)
Vector 55:
      1. announcement 3001
      2. goto step 8 if oldest call-wait in split 1 pri l > 120
      3. goto step 8 if calls-queued in split 1 pri l > 10
      4. queue-to split 1 pri t
      5. wait-time 50 seconds hearing music
      6. announcement 3002
      7. goto step 5 if unconditionally
      8. route-to number 2020 with cov n if unconditionally

  VDN (extension=2020 name=''Message Service''  vector=100)
  Vector 100:
      1. announcement 3900 (''We're sorry, all our
         agents are busy. Please leave a message. Thank you.'')
      2. messaging split 18 for extension 3000
      3. disconnect after announcement 2505 (''We cannot
       take a message at this time. Please call back tomorrow.'')
```

In the first vector, a branch is made to step 8 from step 2 if the condition in the latter step (**`oldest call-wait in split 1 > 120 seconds`**) is true. If the condition is false, a branch is made to step 8 from step 3 if the condition in the latter step (**`calls-queued in split 1 > 10`**) is true. If that condition is also false, the call is queued (step 4), and a wait-announcement loop becomes effective (steps 5 through 7).

If a successful branch to step 8 is made from step 2, the **`route-to number`** command is executed. The destination number in this particular command, 2020, is a VDN. Accordingly, vector processing terminates in the first vector and begins at the first step of the second vector, to which the VDN points.

Once processing control is passed to the second vector, the caller is provided with the appropriate announcement (step 1). Thereafter, upon execution of the **`messaging split`** command in step 2, the system attempts to either queue the call to the message service split or else terminate the call to a message service agent or to an AUDIX voice port. If one of these attempts succeeds, the caller can leave a message. If none of the attempts succeed, the command fails, and vector processing continues at the next vector command. There is usually an announcement that explains to the caller that the necessary connection could not be made.

Service Observing routing

Service Observing vectors allow users to observe calls either from a remote location or from a local station. An example vector that connects a call to a Service Observing feature access code (FAC) is shown in the following example.

Important:

This example does not provide security checks and should be used only in situations where security is not a concern.

Vector for Service Observing FAC

```
1. wait-time 0 secs hearing ringback
2. route-to number #12 with cov n if unconditionally (Listen-only FAC)
3. busy
```

In the example shown above, the caller is connected to a listen-only Service Observing FAC. Once connected, the person who is service observing must dial the extension number that is to be observed. To observe in a listen/talk mode, the observer would dial a different VDN.

For more information about Service Observing see *Administrator Guide for Avaya MultiVantage Software*, 555-233-506, and *Avaya MultiVantage Call Center Software – Guide to ACD Call Center*, 555-230-716.

Branching/Programming commands

Basic Call Vectoring provides programming methods that can be used within a vector either to create branching patterns in call processing flows, or stop vector processing.

Branching/programming commands include:

- goto step and goto vector commands on page 121
- stop command on page 123

goto step and goto vector commands

The **`goto step`** and **`goto vector`** commands can be written to use unconditional branching or conditional branching.

The **`goto`** commands can also be used to connect multiple VDNs. If this command is used in a vector stem to connect to a different VDN, the following events occur:

1. Vector processing continues at the first step in the branched-to vector.
2. Call (if queued) remains in queue.
3. Wait treatment (if any) is continued.
4. Processing then continues in the receiving vector at step 1.

Unconditional branching

Unconditional branching is a method that always passes control from the current vector step to either a preceding or subsequent vector step or to another vector. This type of branching is enabled via the **goto step** and **goto vector** commands, each with a condition of **unconditionally** assigned.

An example of a vector in which unconditional branching is used is shown below.

Unconditional branching example

```
1. goto step 8 if calls-queued in split 3 pri m > 10
2. queue-to split 3 pri m
3. wait-time 12 seconds hearing ringback
4. announcement 3001
5. wait-time 30 seconds hearing music
6. announcement 3002
7. goto step 5 if unconditionally
8. busy
```

In the example shown above, the unconditional branch statement in step 7 establishes a loop between steps 5 through 7. Vector processing within the loop terminates when:

- an agent answers the call
- the system recognizes that the caller abandoned the call

Conditional branching

Conditional branching is a method that conditionally passes control from the current vector step to either a preceding or subsequent vector step or to a different vector. This type of branching is enabled via the **goto step** and **goto vector** commands, each with one of the following conditions assigned and tested: **available-agents**, **staffed-agents**, **calls-queued**, **oldest call-waiting**, or **time-of-day**. When Advanced Vector Routing is enabled, additional conditions can be tested: **rolling-asa**, **counted-calls**, **expected-wait**. See Advanced Vector Routing - EWT and ASA on page 125 for more information. When ANI and II-Digits Routing is enabled, the **ani** and **II-digits** conditions can also be tested with a **goto** command. See, ANI /II-digits routing and Caller Information Forwarding (CINFO) on page 139 for more information. If the command condition is not met, control is passed to the step that follows.

An example vector in which conditional branching is implemented is shown below.

Conditional branching example

```
1. goto vector 100 if time-of-day is all 17:00 to all 8:00
2. goto vector 200 if time-of-day is fri 17:00 to mon 8:00
3. goto step 8 if calls-queued in split 1 pri l > 5
4. queue-to split 1 pri l
5. announcement 4000
6. wait-time 60 seconds hearing ringback
7. goto step 5 if unconditionally
8. busy
```

In the example shown above, a conditional branch test statement appears in steps 1, 2, and 3. If the call is placed during nonbusiness hours, which are defined as 5:00 p.m. to 8:00 a.m., the **goto vector** command in step 1 routes the call to vector 100. However, if the call is placed during business hours, control is passed to step 2, where the **goto vector** command there checks whether the call is placed during the weekend. If this is the case, the call is routed to vector 200. If not, control is passed to step 3, where the **goto step** command checks for the number of calls that are queued to the main split. If the number of calls is greater than five, control is passed to **busy** in step 8. If the number of calls is five or less, the call is queued (step 4). Thereafter, an announcement-wait cycle (steps 5 through 7) is implemented until an agent answers the call or the call is abandoned.

stop command

The **stop** command halts the processing of any subsequent vector steps. If a call is not queued when vector processing stops, the call is dropped and tracked as an “abandon” by the Avaya Call Management System (CMS) and/or BCMS. After the **stop** command is processed, any calls that are already queued remain queued, and any wait treatment is continued. Wait treatments include silence, ringback, system music, or alternate audio or music source.

A vector example that uses the **stop** command is shown below.

Stopping vector processing

```
1. goto step 6 if calls-queued in split 21 pri m > 10
2. queue-to split 21 pri m
3. announcement 4000
4. wait-time 30 seconds hearing ringback
5. stop
6. busy
```

If the **stop** command is reached, the queued caller continues to hear ringback. Also, if the **stop** command in step 5 is executed, step 6 is not executed immediately thereafter. The latter step can be executed only if the **goto** command in step 1 succeeds.

A vector will automatically stop processing when:

- The last step vector step is processed
- 1000 vector steps have been processed
- in vectors that use the **interflow-qpos** LAI conditional, 3000 vector steps have been processed

Considerations

You should keep the following considerations in mind when working with Basic Call Vectoring:

- Make the split queues large enough so that all incoming calls queue and are not dropped. If a queue is too small, a **queue-to split** or a **check split** command might fail to queue a call due to a lack of available queue slots. Accordingly, it is also always a good practice to include in the vector a step that checks a split's queue before queuing occurs and a corresponding step that provides alternate treatment if the queue is full. To check the queue size, you can use a **goto** command (for example, **goto Step 5 if calls-queued in split 20 pri l > 30**). The alternate treatment, which, if needed, is usually accessed by the **goto** command that checks the queue size, can queue the call to a backup split, make an unconditional Look-Ahead Interflow attempt, provide a busy signal, etc.
- A default treatment or a **route-to** destination step should be supplied after a **route-to** command in case the first destination is unavailable.
- Calls should not be queued to an unstaffed split (unless this is intended by the customer) without some alternate treatment.
- Interflow calls should not be permitted to interflow back and forth between a remote switch vector and a local switch. This process could cause a single call to use up all available trunks.
- After an announcement is provided, the audible feedback (such as music) should be re-attached.
- For ease-of-use purposes, each specific vector function or operation should be included in a separate vector and linked via one or more **goto vector** commands.
- In creating a vector, commands can be chosen and arranged in a manner such that answer supervision is delayed as long as possible. This should be done to keep down the service cost.
- The caller should always be provided with initial feedback (usually ringback).
- Direct agent calls merit special attention because such calls can affect call queuing. Although direct agent calls take up a queue slot, they are not always reported as using such a slot on the CMS/BCMS reports (discussed in Appendix H: Call Vectoring/EAS and BCMS/CMS interactions on page 563). For example, a direct agent call is never counted toward the total of queued calls within a split (that is, the **calls-queued** test condition has no effect on this type of call).
- If it is necessary for a caller to hear an entire CONVERSANT script before talking to an agent, the caller should not be queued until after the **converse-on** step is executed.
- Audible feedback should be provided prior to a **converse-on step** whenever a large number of digits are to be outpulsed to the VRU.

Chapter 6: Advanced Vector Routing - EWT and ASA

A number of advanced routing features can be used to enhance conditional routing capabilities of Basic Call Vectoring in order to achieve additional efficiencies in contact center operations. These features include:

Rolling Average Speed of Answer (ASA) – Rolling ASA Routing helps you to make routing decisions that are based on the current average time that it takes for a call to be answered in a split or VDN. In this way, a vector can route a call to the VDN or split where it is likely to be answered most quickly.

Expected Wait Time – EWT routing allows you to make routing decisions based on the time that a caller can expect to wait in queue. This wait time can be predicted for a split or for a call. The EWT can also be passed to a VRU so that a caller can be notified of his or her expected time in queue.

VDN Calls – VDN Calls routing helps you to make routing decisions that are based on the number of incoming trunk calls that are currently active in a VDN. With the VDN Calls conditional, a vector can be used to limit the number of simultaneous calls that are made to a particular VDN. For example, if a service agency is contracted to handle 100 simultaneous calls for a client, calls in excess of that number can be routed to a **busy** step.

Advanced Vector Routing - EWT and ASA includes the following topics:

Command set

The commands used in Advanced Vector Routing are listed in the following table.

Advanced Vector Routing command set

Command category	Action taken	Command
Routing		
	Queue the call to a backup ACD split.	**`check split`**
Branching/programming		
	Go to a vector step. Go to another vector.	**`goto step`** **`goto vector`**

Expected Wait Time (EWT)

EWT routing allows you to make routing decisions based on the time that a caller can expect to wait in queue. This wait time can be predicted for a split or for a call. When predicted for a split, the wait time indicates the amount of time that the caller can expect to wait if the call is queued to the specified split. When predicted for a call, the wait time indicates the time remaining that the caller can expect to wait in queue until the call is serviced from the queue. The EWT can also be passed to a VRU so that a caller can be notified of his or her expected time in queue. The **`expected-wait`** conditional can be used with either the **`goto`** or **`check`** commands.

Call vectoring offers several conditionals that can be used to estimate the time that a caller will be delayed waiting in queue, for example, EWT, rolling ASA and Oldest Call Waiting (OCW). EWT is the most accurate of these conditionals. It takes into account more real-time and historical information than the other predictors. For example, priority level, position in queue, number of working agents, and so forth.

EWT is very responsive to changing contact center conditions. For example, it adjusts instantly to any staffing changes in the split. If an agent moves into or out of auxiliary work mode, the wait time predictions adjust immediately.

EWT does not include the time in a call vector before the call enters a queue. It also does not include the time that the call rings at a telephone after it is removed from the queue.

See When to use wait time predictions on page 129 for a description of when the predictions are most accurate and the circumstances that limit their accuracy.

EWT for a split

The EWT for a split is the time that a new call is expected to remain in queue if it is queued to the split at the specified priority level. It is generally used to determine if a call should be queued to the split.

The vector shown the following figure uses EWT for a split to determine if a call should be queued to that split.

EWT for a split

```
1. goto step 3 if expected-wait for split 1 pri l < 600
2. busy
3. queue-to split 1 pri l
4. announcement 3001
5. wait-time 998 secs hearing music
```

If there are agents available, EWT is zero.

EWT is infinite if:

- There are no logged-in agents.
- All logged-in agents are in AUX work mode.
- The split queue is full.
- There is no split queue and all agents are busy.
- The split queue is locked (This occurs when the last working agent in a non-vector-controlled split attempts to go into AUX work mode.).

EWT for a call

EWT for a call is the remaining time that a caller can expect to wait before his or her call is serviced from the queue. If the call is queued to multiple splits, the remaining queue time for each of the splits is calculated, and the shortest of these is taken as the call's EWT.

For a call to have an expected wait time it must be queued to at least one split. If it is not queued, or if it is queued to splits that are not staffed, the EWT value is infinite.

An example vector in which EWT is used to determine call treatment is shown in the following example.

EWT for a call

```
1. queue-to split 1 pri m
2. check split 2 pri m if expected-wait < 30
3. goto step 5 if expected-wait for call < 9999
4. busy
5. announcement 3001
6. wait-time 998 secs hearing music
```

Passing EWT to a VRU

The EWT for a call can be passed to a VRU so that a caller can be notified of his or her expected time in the queue. EWT is passed to the VRU with the **`converse-on`** command as “wait” data. The value that is outpulsed to the VRU is the expected wait time of the call in seconds. The VRU can then convert the seconds to a spoken message. The expected wait is calculated after the VRU port answers the call, so queuing to a converse split does not adversely impact the EWT value that is passed to the VRU.

No zero padding is added to the wait time that is passed to the VRU. If the EWT for the call is 128 seconds, the digits 1, 2, and 8 are outpulsed. If the EWT is 5 seconds, the digit 5 is outpulsed.

The wait time that is passed to the VRU is the most accurate prediction possible. On the average, 50% of the time the actual wait time will be shorter and 50% of the time it will be longer. It is recommended that VRU applications make an upward adjustment of the prediction so that the majority of callers receive a predicted wait time that is equal to or greater than their actual wait time.

The VRU can also announce EWT to a caller periodically throughout the time that a call is in queue. In this way, the caller can observe his or her progress up the queue. However, this approach should be used with caution. Circumstances such as a reduction in the number of agents or a sudden influx of higher priority calls could cause the caller’s EWT to increase from one announcement to the next.

If the call is not queued or if it is queued only to splits that are unstaffed or splits where all agents are in AUX work mode, the end-of-string character “#” is the only data item that is outpulsed.

The EWT algorithm

EWT is calculated using an algorithm that is based on the number of calls in a queue at a particular priority level and the rate of service of calls from the queue at that priority level. It adjusts for many other factors such as multiple split queuing, call handling times, and the impact of direct agent calls on the wait time of other calls to the split. The algorithm adjusts EWT immediately for changes in staffing, such as agents logging in or taking breaks in AUX work mode.

Changes occur constantly in a contact center and EWT cannot predict the future. Therefore, the accuracy of the EWT predictions are in proportion to the rate at which calls are serviced from the queue and the level of stability that is achieved in the contact center between the time that the prediction is made and the time that the call is serviced from queue.

When to use wait time predictions

This section contains a number of situations that can have an adverse impact on the accuracy of wait time predictions. These factors have an adverse impact on all predictors, not just on EWT. The EWT algorithm should still be more accurate than other predictors, even when these situations are present.

Wait time predictions are best suited for medium-volume or high-volume call scenarios. In general, the potential accuracy of a wait time predictor increases as the rate of removal from queue increases. It is recommended that EWT be used when the rate of removal from queue at a given split priority level is at least one call every 30 seconds.

Predictions can be made for a split with multiple priority levels in use as long as the majority of calls are delivered to the lower priority levels. If the majority of calls are queued at the higher-priority levels, any predictions made for the lower-priority levels may not be accurate.

The following list describes circumstances that limit the accuracy of the wait time predictions.

- Immediately after a system restart or when a new split is administered.

  The EWT algorithm uses a combination of historical and real-time information to make predictions. When no historical information exists, such as when a new split is added or a reset system 3 or 4 is completed, there is the potential for inaccuracies.

  To prevent inaccurate predictions when there is no historical information, administer the Expected Call Handling Time field on the Hunt Group form. The value in this field is then used in place of the missing historical data.

  If the value of this field does not accurately reflect the call handling times of the split, EWT predictions may be inaccurate until some call history is generated. The algorithm normally requires about 30 queued calls to be answered from a split priority level before it reaches its maximum accuracy.

  You can change the value in the “Expected Call Handling Time” field by executing a change hunt group command. Changing the value does not disrupt EWT predictions by overwriting EWT history. The value is stored and used the next time a reset system 3 or 4 is executed.

- Low call volume applications.

  Split priority levels where the rate of removal from the queue is very low can only be predicted with limited accuracy.

- Sites with frequent staffing changes.

  Although EWT immediately adjusts for all types of staffing changes, since predictions may have already been made for calls that are waiting in queue, those past predictions were based on staffing information which is now out of date. Therefore, the EWT in scenarios where large staffing changes are continually happening can only be predicted with limited accuracy.

- Staffed agents who rarely answer calls to a split.

  The EWT algorithm takes account of agents in multiple splits in its calculation. However, suppose there are many agents who are assigned to a split but spend most of their time answering calls in their other splits. If a large number of these agents are moved to or from the split, the EWT for this split may be temporarily inaccurate until it adjusts to those changes.

- Applications with widely varying call handling times.

  If the majority of calls to a split are handled within a narrow range of times, the accuracy of any predictor will be much greater than that for a split where call handling times are widely different.

EWT routing and passing wait to a VRU

The following example shows routing that is based on the wait time of a split, as well as passing wait data to the VRU. Wait time is only given to the caller if the caller is expected to wait a total of more than 60 seconds in queue. Callers who would wait more than 10 minutes are told to call back later.

Using EWT and VRU routing to pass wait time to callers

```
1. goto step 3 if expected-wait for split 32 pri l < 600
2. disconnect after announcement 13976
3. queue-to split 32 pri l
4. wait-time 20 secs hearing ringback
5. goto step 7 if expected-wait for call < 40
6. converse-on split 80 pri l passing wait and none
7. announcement 11000
8. wait-time 60 secs hearing music
9. goto step 7 if unconditionally
```

Calls with more than 10 minutes to wait fail step 1 and are disconnected after an announcement that is asking them to call back later. If the expected wait time is less than 10 minutes step 1 routes the call to step 3 where it is queued to split 32 and waits 20 seconds hearing ringback. After 20 seconds if the expected wait time for the call is less than 40 seconds, step 5 routes the call to an announcement followed by a wait with music. If the expected wait time for the call is equal to or greater than 40 seconds, step 6 informs the caller of the amount of time that he or she can expect to wait before the call is answered.

Notifying callers of wait time without a VRU

You can use EWT to notify callers of their expected wait time without a VRU. This can be done using recorded announcements and by associating each recorded announcement with a time band as shown in the following example.

Using EWT and announcements to pass wait time to a caller

```
VECTOR 101
1.   queue-to split 3 pri h
2.   goto step 4 if expected-wait for call <= 600
3.   busy
4.   wait-time 12 seconds hearing ringback
5.   announcement 3001 (“Thank you for calling ABC Inc. All agents
     are busy, please wait and we will get to your call as soon as
     possible”)
6.   goto vector 202 if unconditionally

_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

VECTOR 202
1.   goto step 13 if expected-wait for call > 280
2.   goto step 11 if expected-wait for call > 165
3.   goto step 9 if expected-wait for call > 110
4.   goto step 7 if expected-wait for call > 55
5.   announcement 3501 (“Thank you for waiting.
     Your call should be answered within the next minute”)
6.   goto step 14 if unconditionally
7.   announcement 3502 (“Thank you for waiting. Your call should be
     answered within approximately one to two minutes”)
8.   goto step 14 if unconditionally
9.   announcement 3503 (“Thank you for waiting. Your call should be
     answered within approximately two to three minutes”)
10.  goto step 14 if unconditionally
11.  announcement 3504 (“Thank you for waiting. Your call should be
     answered within approximately three to five minutes”)
12.  goto step 14 if unconditionally
13.  announcement 3505 (“We apologize for the delay. Due to heavy
     call volume, you may have to wait longer than five minutes
     to speak to a representative. If possible, we suggest that you
     call between the hours of 8am and 10am for the fastest service”)
14.  wait-time 120 secs hearing music
15.  goto step 1 if unconditionally
```

In step 1, the call is queued to split 3 at high priority. If the call fails to get a queue slot in split 3, if split 3 has no working agents, or if the wait time in split 3 at high priority exceeds 10 minutes, step 2 fails and the caller receives a busy signal. If step 2 succeeds, the caller hears ringback and an announcement and is then sent to vector 202. Steps 1 through 4 of vector 202 determine which of five time bands the caller’s remaining queuing time is estimated to be within. One of five recorded announcements is then played to the caller to inform him or her of the expected wait time.

Notice that the EWT thresholds are set lower than the times that are quoted in the recorded announcements. Callers may become upset if their actual wait time exceeds the time stated in the announcement. Therefore, you may want to program your vectors such that few callers ever experience wait times that exceed the wait time of the announcement.

Notice also that vector 202 can be used for any application that requires that calls be notified of their remaining time in queue.

Using EWT to route to the best split

With EWT, you may want to change the normal queuing strategy of queuing calls to multiple splits to ensure that the call is answered in the shortest possible time. This strategy uses additional system resources and can make it more difficult to read and analyze split reports.

Instead, you may want to use EWT to determine up-front which split is best for each call and avoid multiple split queuing.

The following example shows a scenario that includes a main split (1) and a backup split (2). Either split can service a particular type of call. It is preferable that an agent from the main split service the call. However, a 30-second maximum wait time is also desirable.

The strategy in this vector is to use the backup split only if the backup split can answer the call within 30 seconds and the main split cannot.

EWT routing to the best split

```
1.   goto step 5 if expected-wait for split 1 pri m <= 30
2.   goto step 5 if expected-wait for split 2 pri m > 30
3.   check split 2 pri m if unconditionally
4.   goto step 6 if unconditionally
5.   queue-to split 1 pri m
6.   wait-time 12 secs hearing ringback
7.   announcement 3501
8.   converse-on split 18 pri m passing wait and none
9.   wait-time 120 secs hearing music
10. goto step 8 if unconditionally
```

Step 1 branches to step 5 to queue to the main split if the main split can answer the call within 30 seconds. If the main split cannot answer the call within 30 seconds, step 2 checks to see if the backup split can answer the call within 30 seconds. If it cannot, the call branches to step 5 and is queued to the main split. If it can, the call is queued to the backup split in step 3. At this point, the call is queued either to the main split or to the backup split but not to both.

Steps 6 through 10 provide audible feedback to the caller while the call is in the queue. Note that in step 8, which is executed every 2 minutes, a VRU is used to provide the caller with his or her remaining wait time.

Factors that influence EWT values

The value that is returned to the switch as the expected wait time can vary depending on a variety of conditions, as discussed below.

Factors that increase EWT for a split priority level

The most common causes for an increase in EWT for a split priority level are:

- The number of calls that are in queue increases
- Agents log out
- Agents go on break or are otherwise in the AUX work mode
- Agents are moved to another split
- Agents with multiple splits answer an increasing number of calls in other splits

Other conditions that may also cause EWT for a split priority level to increase include:

- The average talk time increases
- The number of calls at a higher priority increases
- The number of Direct Agent calls increases
- The number of RONA calls increases
- The number of abandoned calls decreases
- The number of calls that are queued in this split but answered in another decreases.

Factors that decrease EWT for a split priority level

The most common causes for a decrease in EWT for a split priority level are:

- The number of calls in queue decreases
- Agents log in (and start answering calls)
- Agents return from break or otherwise are no longer in the AUX work mode
- Agents are moved from another split
- Agents with multiple splits answer fewer calls in other splits

The following conditions may also cause a decrease in EWT for a split priority level:

- The average talk time decreases
- The number of calls at higher priority decreases
- The number of Direct Agent calls decreases
- The number of RONA calls decreases
- The number of abandoned calls increases
- The number of calls queued in this split but answered in another increases

Troubleshooting EWT

To verify that your EWT is operating as intended, use the **`list trace ewt`** command to observe processing events of all calls.For more information, see Appendix D: Troubleshooting vectors on page 493.

Rolling Average Speed of Answer (ASA)

Rolling ASA Routing helps you to make routing decisions that are based on the current average time that it takes for a call to be answered in a split or VDN. In this way, a vector can route a call to the VDN or split where it is likely to be answered most quickly.

The Average Speed of Answer that is used for vector routing is called “rolling” ASA to differentiate it from the “interval” ASA that is recorded in Basic Call Management System (BCMS) and Avaya Call Management System (CMS) reports. Rolling ASA is a running calculation that does not take into account the 15-minute, half-hour, or hour reporting intervals. It does not reflect interval boundaries. The “interval” that ASA uses for reporting is calculated on reporting interval boundaries and clears to zero at the start of each reporting interval.

The rolling ASA for a split or VDN is calculated based on the speed of answer for all calls recorded since system start-up. When rolling ASA is calculated, each call is given a weighted value that is greater than the call that preceded it. In this way, the most recent calls contribute the most to the average. Approximately 95% of the value of rolling ASA is obtained from the last ten calls.

The rolling ASA for a split or VDN is recalculated every time that a call is answered so that it always reflects the most recently available data. Calls that are not answered, for example calls that receive a forced busy, are not considered for the rolling ASA calculation.

The rolling ASA is calculated for an entire split or VDN. The calculation does not consider the priority levels of answered calls.

The following sections explain what is included in the rolling ASA calculation for a split or VDN.

Rolling ASA split calculation

The rolling ASA for a split is the average time that it takes for a call to be answered from the time that the call attempts termination to the split until it is answered in that split. Rolling ASA includes the time that the call is waiting in the queue and the time that it is ringing at a telephone.

If the call is answered in another split or the call is abandoned by the caller before it is answered, rolling ASA is not recorded for the call. If a call flows into a split from another split, the time queued and ring time for the previous split are not included. If a call is queued in multiple splits, only the rolling ASA for the split in which the call is answered is impacted.

Rolling ASA VDN Calculation

The rolling ASA for a VDN is the average time that it takes for a call to be answered from the time that it starts processing within the specified VDN until it is answered. It includes any time that is spent in vector processing, including the amount of time that is spent in announcements that are administered as vector steps. If the call is answered by an agent, it includes the time that the call is waiting in the queue and the time that it is ringing at the agent’s telephone.

The rolling ASA for a VDN only includes data from calls that are answered in that VDN. If a call flows between VDNs, only the time that is spent within the answering VDN is used in the calculation. For example, if a call is placed to VDN1, after ten seconds routes to VDN2, and is then answered in VDN2 after five seconds, the ASA for the call is recorded in VDN2 as five seconds. Nothing is recorded for VDN1 since the call was not answered there.

The VDN for a vector step can be specified in three ways:

- A VDN number
- The value “latest”. The “latest” VDN is the VDN that is currently processing the call. The value is not affected by VDN override.
- The value “active.” The “active” VDN is the VDN of record. That is, it is the called VDN as modified by override rules. For example, if a call routes from a VDN with override set to **yes** then the new VDN is the active VDN. If a call routes from a VDN with override set to **no,** the previous VDN is the active VDN.

Rolling ASA considerations

Because of its greater accuracy and greater flexibility, EWT is recommended over rolling ASA as a predictor of split/skill waiting time. However, rolling ASA is provided for those who may have a special requirement or want to use the more traditional ASA measurement.

Normally, rolling ASA conditionals should not be used to prevent from calls queuing to the main split/skill or being answered in the principal VDN. Rather, rolling ASA should be used to see whether vector processing should attempt to queue the call to additional splits/skills if the main split/skill does not currently meet the targeted threshold. If no calls are being answered in the main split/skill or VDN, the value of rolling ASA does not change. This could result in all future calls being locked out of the main split/skill or VDN unless there are other call vectors in the system that are directing calls to them.

If you want to implement a call flow that decides whether or not to queue a call to a main split/skill, use the EWT feature.

Combining VDN and ASA routing

The following figure shows an example vector that combines VDN and split ASA routing.

Rolling ASA routing

```
1.   queue-to split 10 pri h
2.   goto step 6 if rolling-asa for split 10 <= 30
3.   check split 11 pri h if rolling-asa <= 30
4.   check split 12 pri h if rolling-asa <= 30
5.   check split 13 pri h if rolling-asa <= 30
6.   announcement 10000
7.   wait-time 40 secs hearing music
8.   goto step 3 if unconditionally
```

Step 1 queues the call to the main split. If the main split is currently answering calls within the target time of 30 seconds, step 2 bypasses all of the backup splits and goes directly to the announcement in step 6. The assumption is that the call will be handled by split 10 within the time constraints. However, if the call is not answered by the time that vector processing reaches step 8, the backup splits are checked.

If the rolling ASA for the main split is greater than 30 seconds, steps 3, 4, and 5 check the backup splits. The call is queued to any of these splits that have a rolling ASA of 30 seconds or less. If the call still is not answered by the time that vector processing reaches step 8, the backup splits are checked again.

VDN Calls

VDN Calls routing helps you to make routing decisions based on the number of incoming trunk calls that are currently active in a VDN. With the VDN Calls conditional, a vector can be used to limit the number of simultaneous calls that are made to a particular VDN. For example, if a service agency is contracted to handle 100 simultaneous calls for a client, calls in excess of that number can be routed to a **`busy`** step.

When Advanced Vector Routing is enabled, a count of active incoming trunk calls is kept for each VDN. The VDN counter is incremented each time that an incoming call is placed to the VDN. It is decremented each time that an incoming call is released. A call is considered active in a VDN from the time the call routes to the VDN until all parties on the call are dropped and the call is released.

Note:

The call is counted for the originally called VDN only. When a call is routed to another VDN, the call counter for the subsequent VDN is not incremented, nor is the call counter for the original VDN decremented.

As with other Advanced Vector Routing conditionals, the VDN for a **`goto`** step can be specified in three ways: a VDN number, the value “latest,” or the value “active.”

The following section describes which calls are included in the VDN Calls counts and which are not.

Counted calls

The VDN call count includes:

- Incoming trunk calls that are routed directly to the VDN
- Incoming trunk night service calls where the VDN is the night service destination
- Calls that cover or forward to the VDN if it is the first VDN routed to and the call is an incoming trunk call
- Already counted calls that are conferenced with counted or not counted calls from the same VDN

The VDN call count does not include:

- Internal calls to the VDN
- Calls that are transferred to the VDN
- Calls that are redirected to their VDN return destination
- Conferenced calls that were previously counted on different VDNs

Using the counted-calls conditional

The following figure shows an example vector shows how the **counted-calls** conditional can be used to route calls.

Using VDN call counting to route calls

```
1.   goto step 3 if counted-calls to vdn 1234 <= 100
2.   busy
3.   queue-to split 60 pri l
4.   wait-time 20 seconds hearing ringback
5.   announcement 27000
6.   wait-time 60 seconds hearing music
7.   goto step 5 unconditionally
```

If more than 100 calls are active in VDN 1234, the caller hears a busy signal and vector processing is terminated. If 100 or fewer calls are active, the call queues to split 60.

Chapter 7: ANI /II-digits routing and Caller Information Forwarding (CINFO)

The ANI (Automatic Number Identification) and II-digits (Information Indicator Digits) Call Vectoring features help you to make vector routing decisions that are based on the caller identity and the type of the originating line. Caller Information Forwarding (CINFO) makes it possible for you to collect caller entered digits (ced) and customer database provided digits (cdpd) for a call from the network.

When ANI and II-digits are provided with an incoming call to a VDN, they are sent to Avaya Call Management System (CMS) when vector processing starts. ANI, II, and CINFO digits are forwarded with interflowed calls. In addition, ANI and II-digits are passed over ASAI in event reports.

ANI /II-digits routing and Caller Information Forwarding (CINFO) includes the following topics:

Command sets

The following table lists the commands that are used by ANI, II-digits, and CINFO digits.

Command category	Action taken	Command
Branching / Programming		
	Go to a vector step (ANI, II-digits). Go to a vector step that is based on ced or cdpd (CINFO digits).	**`goto step`**
	Go to another vector (ANI, II-digits). Go to another vector based on ced or cdpd. (CINFO digits).	**`goto vector`**
Information Collection		
	Pass ANI to a Voice Response Unit. Pass ced and cdpd to a Voice Response Unit (CINFO).	**`converse-on`**
	Collect ced and cdpd from a network ISDN SETUP message.	**`collect digits`**
Routing		
	Route the call to a number that is programmed in the vector, based on ced or cdpd.	**`route-to number`**
	Route the call to digits supplied by the network.	**`route-to digits`**
	Request routing information from an ASAI adjunct that is based on ced or cdpd.	**`adjunct-routing`**

ANI routing

ANI routing helps you to make routing decisions based on incoming or internal caller identity. Calls from a specified customer can receive unique routing, local calls can be routed differently from long distance calls, or calls from different geographical areas can receive different routing. ANI also can be compared against entries in a Vector Routing Table.

ANI is based on the Calling Party Number (CPN). It is not always identical to the Billing Number. For example, if the call is placed by a user from a switch, the CPN can be either the switch-based billing number or the station identification number.

The ANI routing digit string can contain up to 16 digits. This supports international applications. However, ANI information in North America contains only 10 digits.

The following calls have ANI values associated with them:

- Incoming ISDN-PRI calls that send ANI
- Incoming R2MFC signaling calls that send ANI
- DCS calls
- Internal calls

If ANI is not provided by the network for a call, ANI is available for vector processing on that call.

When an EAS agent makes a call to a VDN, the agent’s login ID is used as the ANI instead of the number of the physical terminal.

When a call is transferred internally to a VDN, the following is true:

- If the transfer is completed before the call reaches the ANI conditional, the ANI value of the originator of the call is used.
- If the transfer is completed after the call reaches the ANI conditional, the ANI value of the terminal that executes the transfer is used.

To ensure that the originator’s ANI is preserved during a transfer, add a filler step (such as wait with silence) to the beginning of the vector. In this way, a transfer can be completed before the ANI conditional is encountered.

The ANI value that is specified for a goto step can include the “+” and/or the “?” wildcards. The “+” represents a group of zero or more digits and can be used only as the first or last character of the string. The “?” represents a single digit. Any number of the wildcard can be used at any position in the digit string.

ANI routing example

The vector example shown below provides several applications of ANI Routing.

ANI routing example

```
1.   wait-time 4 secs hearing silence
2.   goto step 13 if ani = none
3.   goto step 12 if ani = 3035367326
4.   goto vector 74920 if ani <= 9999999
5.    goto vector 43902 if ani = 212+
6.   goto vector 43902 if ani = 202+
7.   wait-time 0 seconds hearing ringback
8.   queue-to split 16 pri m
9.   wait-time 120 seconds hearing 32567 then continue
10.  announcement 32456
11.  goto step 9 if unconditionally
12.  route-to number 34527 with cov y if unconditionally
13.  route-to number 0 with cov n if unconditionally
14.  busy
```

In step 2, calls that do not have ANI associated with them are routed to an operator. Step 3 routes calls from a specific telephone to a specified extension. Step 4 routes local calls, which are calls with 7 or fewer digits, to a different vector. Steps 5 and 6 route calls from area codes 212 and 202 to a different vector. Calls that are not rerouted by the previous steps are then queued.

Vector routing tables with ANI

You can also test ANI against entries in a Vector Routing Table.

Vector Routing Tables contain lists of numbers that can be used to test a **`goto...if ani`** command. ANI can be tested to see if it is either in or not-in the specified table. Entries in the tables can also include the “+” and/or “?” wildcard.

The example Vector Routing Table shown below includes various area codes for the state of California.

Vector routing table for ANI routing

```
                   VECTOR ROUTING TABLE
  Number: 6              Name: California          Sort? n


1:  714+                                   17: _______
2:  805+                                   18: _______
3:  619+                                   19: _______
4:  707+                                   20: _______
5:  209+                                   21: _______
6:  310+                                   22: _______
7:  213+                                   23: _______
8:  408+                                   24: _______
9:  510+                                   25: _______
10: 818+                                   26: _______
11: 909+                                   27: _______
12: 916+                                   28: _______
13: 415+                                   29: _______
```

The vector example shown below could be used to route area code calls to a different vector.

Testing for ANI in vector routing table

```
1. announcement 45673
2. goto step 9 if ani = none
3. goto vector 8 if ani in table 6
4. queue-to split 5 pri l
5. wait-time 10 seconds hearing ringback
6. announcement 2771
7. wait-time 10 seconds hearing music
8. goto step 6 if unconditionally
9. route-to number 0 with cov y if unconditionally
```

In the example vector shown above, if no ANI is available for the call, it is routed to an operator. If the first three numbers match an area code from table 6, the call is routed to vector 8. All other calls are queued.

II-digits routing

II-digits routing helps you to make routing decision based on the type of the originating line. In this way, calls from pay telephones, cellular telephones, or motel telephones, for example, can receive unique routing.

II-digits is a 2-digit string that is provided for an incoming call by ISDN PRI. II-digits delivery is a generally available ISDN PRI AT&T Network service. This service is bundled with ANI delivery and tariffed under the MEGACOM 800® and MultiQuest 800® INFO-2 feature to provide information about the call's origination. II-digits indicates the type of originating line. R2-MFC Call Category digits, when available, are treated as II-digits for routing.

The II-digits routing string can only contain two characters. The string can contain either the "+" or "?" wildcard. Leading zeros are significant. For example, the II-digits value "02" that is associated with a call will not match the digit string "2" in a vector step.

As with ANI routing and collected-digit routing, II-routing digits can be compared against entries in a Vector Routing Table.

When a call is returned to vector processing as a result of the VDN Return Destination feature, the II-digits are preserved.

Call types associated with II-digits

The following calls have II-digits values associated with them:

- Incoming ISDN PRI calls that include II-digits
- Incoming ISDN PRI Tie Trunk DCS or non-DCS calls that include II-digits

Note:

Since tandeming of II-digits is only supported if the trunk facilities used are ISDN PRI, traditional DCS does not support II-digits transport but DCS Plus (DCS over PRI) does.

Possible uses for II-digits

Some possible uses for II-digits routing include:

- Help detect fraudulent orders for catalog sales, travel reservations, money transfers, traveler's checks, and so forth
- Assign priority or special treatment to calls that are placed from pay telephones, cellular telephones, or other types of lines. For example, special priority could be given by an automobile emergency road service to calls that are placed from pay telephones
- Detect calls placed from pay telephones when it is the intention of the caller to avoid being tracked by collection agencies or dispatching services
- Convey the type of originating line on the agent display by routing different type calls to different VDNs

II-digit states for transferred calls

When a call is transferred internally to a VDN, the following is true:

- If the transfer is completed before the call reaches the II-digits conditional, the II-digits value of the originator of the call is used.
- If the transfer is completed after the call reaches the II-digits conditional, the II-digits value of the terminal that is executing the transfer is used. Under normal circumstances, there are no II-digits for a terminal that executes a transfer.

To ensure that the originator's II-digits is preserved, add a filler step such as **`wait with silence`** to the beginning of the vector. In this way, a transfer can be completed before the II-digits conditional is encountered.

The following table provides a summary of currently available II-digits. A complete and more descriptive list of II-digits is published quarterly in Section 1 of the "Local Exchange Routing Guide" published by Bellcore.

II-digits summary

Code	Use
00	Identified line - no special treatment
01	Multiparty - ANI cannot be provided
02	ANI failure
06	Hotel/Motel - DN not accompanied by automatic room ID
07	Special operator handling required
20	AIOD - Listed DN of PBX sent
23	Coin or Non-Coin - line status unknown
24	800 Service
27	Coin Call
29	Prison/Inmate Service
30-32	Intercept
34	Telco Operator Handled Call
40-49	Locally determined by carrier
52	OutWATS
60	Telecommunication Relay Service (TRS) - Station Paid
61	Type 1 Cellular

II-digits summary (continued)

Code	Use
62	Type 2 Cellular
63	Roaming Cellular
66	TRS - From Hotel/Motel
67	TRS - From restricted line
70	Private paystation
93	Private Virtual Network call

II-digits routing example

The following example shows branching calls with different II-digits to different VDNs. The VDN override is set to “yes” on the called VDN. In this way, the VDN name or VDN of Origin Announcement can be used to convey to the agent the type of II-digits that are associated with the call.

II-digits routing example

```
1. goto step 9 if ii-digits = none
2. goto step 10 if ii-digits = 00
3. goto step 11 if ii-digits = 01
4. goto step 12 if ii-digits = 06
5. goto step 13 if ii-digits = 07
6. goto step 13 if ii-digits = 29
7. goto step 14 if ii-digits = 27
8. goto step 15 if ii-digits = 61
9. route-to number 1232 with cov n if unconditionally
10. route-to number 1246 with cov n if unconditionally
11. route-to number 1267 with cov n if unconditionally
12. route-to number 1298 with cov n if unconditionally
13. route-to number 1255 with cov n if unconditionally
14. route-to number 1298 with cov n if unconditionally
15. route-to number 1254 with cov n if unconditionally
```

In this vector, step 1 routes calls with no associated II-digits to extension 1232. Steps 2 through 8 route calls with different II-digits to different extensions.

Caller Information Forwarding

Caller Information Forwarding (CINFO) makes it possible for you to use **`collect digits`** steps to retrieve caller entered digits (ced) and customer database provided digits (cdpd) that are supplied by the network in an incoming call's ISDN PRI SETUP message. These network-provided digits are available with AT&T Network Intelligent Call Processing (ICP) service. ISDN-PRI is required.

For example, a caller could dial a number that resulted in ICP routing at the network switch. The network switch could request information from the caller (ced) and/or from the call center customer host database (cdpd). These digits are sent in the call ISDN message to the switch and are then available for a **`collect digits`** vector steps.

Up to 30 ced and/or up to 30 cdpd can be stored for a call. These digits also are forwarded with a call that is tandemed or interflowed.

Detailed operation

When an ISDN call is received from either the AT&T network or a tandemed PRI call, the system stores the Codeset 6 User Entered Code (UEC) Information Element when it contains the ced and/or cdpd. If more than one ced UEC IE is received, only the first one is stored or tandemed with the call. If more than one cdpd UEC IE is received, only the first one is stored or tandemed with the call.

When a **`collect ced digits`** or **`collect cdpd digits`** step is processed, the system retrieves the ced or cdpd and places them in the collected digits buffer. Any digits that were in the collected digits buffer, such as dial-ahead digits, are erased. If a TTR was connected to the call from a previous **`collect digits`** step, the TTR is disconnected.

If the ced or cdpd contain invalid digits, the system does not store the UEC IE. Valid digits are 0 through 9, *, and #. When the **`collect digits`** step is reached, the collected digits buffer is still cleared and if a TTR is attached, it is still disconnected. A vector event is generated that indicates that no digits were collected.

If no ced or cdpd were received from the network, when the **`collect ced digits`** or **`collect cdpd digits`** step is reached, the step is skipped. However, the collected digits buffer is still cleared and if a TTR is attached, it is still disconnected.

If an asterisk (*) is included in the collected digits, it is treated as a delete character. Only the digits to the right of the asterisk are collected. If a pound sign (#) is included in the collected digits it is treated as a terminating character. Only the pound sign and the digits to the left of it are collected. If a single pound sign is sent, it is placed in the collected digits buffer.

The number of ced or cdpd to collect cannot be specified in the **`collect digits`** step. If there are 16 or fewer digits, all the digits are collected. If there are more than 16 digits, the first 16 digits are collected and a vector event is generated.

The CINFO ced and cdpd can be used with any vector step that uses the digits in the collected digits buffer. These steps are:

- **`adjunct routing`** (digits passed in an event report as collected digits)
- **`converse-on...passing digits`**
- **`goto...if digits...`**
- **`goto...if digits in table...`**
- **`route-to digits`**
- **`route-to number ... if digit...`**

In the same way as other collected digits, ced or cdpd can be displayed using the CALLR INFO button on the telephone.

When a call is transferred internally to a VDN, the following is true:

- If the transfer is completed before the call reaches the CINFO conditional, the CINFO value of the originator of the call is used.
- If the transfer is completed after the call reaches the CINFO conditional, the CINFO value of the terminal that executes the transfer is used.

To ensure that the originator's CINFO is preserved during a transfer, add a filler step such as wait with silence to the beginning of the vector. In this way, a transfer can be completed before the CINFO conditional is encountered.

To retrieve both the ced and cdpd for a call, you must use two **`collect digits`** steps. Because the **`collect digits`** command for ced or cdpd clears the collected digits buffer, the ced or cdpd that is collected first must be used before the second set is requested. The following sample vector shows an application where both ced and cdpd are used.

CINFO vector example

In the following example ced and cdpd are both used to determine routing for the call.

CINFO example

```
 1. wait-time 2 secs hearing silence
 2. collect ced digits
 3. goto step 7 if digits = 1
 4. goto step 11 if digits = 2
 5. route-to number 0 with cov n if unconditionally
 6. stop
 7. collect cdpd digits
 8. route-to digits with coverage n
 9. route-to number 0 with cov n if unconditionally
10. stop
11. queue-to split 6 pri m
12. wait-time 10 secs hearing ringback
13. announcement 2564
14. wait-time 20 secs hearing music
15. goto step 13 if unconditionally
16. route-to number 0 with cov n if unconditionally
```

In this vector, step 1 provides a wait-time step in case calls will be transferred to this vector. Step 2 collects the ced. Steps 3 and 4 branch the call to a different vector step depending on the ced digit that was received. If no ced were received, or if the digit received was not 1 or 2, step 5 routes the call to the attendant. If the ced digit collected was 1, the call routes to a second collect step where cdpd are collected. The vector then routes the call to the cdpd. If the ced digit collected was 2, the call queues to split 6.

CINFO interactions

The following paragraphs discuss the interaction of CINFO with other features and applications.

- ASAI

  ced and cdpd can be passed to an ASAI adjunct as collected digits with the adjunct routing command and other event reports. ASAI will pass a maximum of 16 digits.

  If a TTR is connected to a call as a result of ASAI-Requested Digit Collection, and the call encounters a collect ced or cdpd step, the TTR is disconnected from the call. In addition, any ASAI-requested digits that are stored in the collected digit buffer are discarded and no entered digits event report is sent.

  ASAI does not distinguish between CINFO digits and user-entered digits that are collected as a result of a **`collect digits`** step. In other words, CINFO digits are provided to an ASAI adjunct but without any indication that they are anything other than collected digits from a vector.

  The Call Offered to (VDN) Domain Event Report will contain the digits from the most recent **`collect ced`** or **`collect cdpd`** vector step.

- Best Service Routing (BSR)

  BSR digits are included with the call if a multi-site BSR application routes the call to another switch.

- Avaya CMS

  The Vectoring (CINFO) customer option is not required for ced or cdpd to be passed to CMS. Any version of the CMS will accept ced or cdpd.

- Conference

  When a conference is established, the CINFO digits are merged into the call record of the conference. However, there is no indication of which party the digits originally belonged with. Therefore, for security reasons, when the first ISDN call drops out of the conference, the CINFO digits are erased.

- Look-Ahead Interflow

  CINFO digits are included with the call if Look-Ahead Interflow routes the call to another switch. The collect ced/cdpd step is neutral for Look-Ahead Interflow.

- Transfer

  If a call is transferred off the DEFINITY, the CINFO digits are lost. If a call is transferred to an internal extension, CINFO digits are retained.

  If a call is transferred to a VDN, the CINFO digits should not be collected until the transferring party has had time to complete the transfer. If transfers are likely, a wait-time step of sufficient length is recommended before the collect step.

Chapter 8: Information Forwarding

Standard Information Forwarding involves the transport of the following incoming call-related information:

- ANI
- II-Digits
- CINFO
- ASAI-provided user information
- Look-Ahead Interflow (LAI) information, such as VDN name

The switch also supports Information Forwarding for:

- Universal Call ID (UCID)—Provides a unique identifier for each call that is used to track the call. For more information, see Universal Call ID in *Administrator Guide for Avaya MultiVantage Software*, 555-233-506.
- Best Service Routing (BSR)—Allows the switch to compare specified splits or skills, identify the split or skill that will provide the best service to a call, and deliver the call to that resource. For more information, see Best Service Routing (BSR) on page 229.
- New interflowed call data including collected digits and in-VDN time

Note:

This transport takes place by way of globally-supported ISDN information transport methods over public and private networks using ISDN trunks. For example, PRI or BRI. Private networks can be configured for QSIG or non-QSIG protocols.

ATM trunking and IP trunking can be set up to emulate ISDN PRI. For information on setting this up, see *Administration for Network Connectivity for Avaya MultiVantage Software*, 555-233-504, and *ATM Installation, Upgrades and Administration for Avaya DEFINITY Servers*, 555-233-124.

When a call is interflowed to an alternative switch by the BSR or Look-Ahead Interflow features, the following data forwarding is supported:

- Collected Digits—Any digits that are collected for the call are passed with the interflowed call, and automatically collected when the call enters vector processing at the receiving switch.
- Elapsed in-VDN time—The elapsed time that the call has already spent at the sending switch is passed with the interflowed call and automatically sent to the Avaya Call Management System (CMS) when the call enters vector processing at the receiving switch.
- UCID—Universal Call ID.

Benefits of Enhanced Information Forwarding

Enhanced Information Forwarding provides the following benefits:

- **Improved agent efficiency and service to call**—forwarding of original caller service requirements and entered prompted digits speeds service to the caller and saves the agent time.
- **Better network-wide call tracking**—forwarding of UCID, In-VDN-Time and collected digits allows tracking as a single call and provides a network-wide view for call statistics.
- **Better CTI integration**—forwarding of UCID, In-VDN-Time, and collected digits provides screen pop and database access applications across sites.
- **Improved global compatibility and viability**—use of codeset 0 supports information transport over ISDN PRI/BRI facilities (QSIG or non-QSIG) as well as supporting operation over public networks.

The following table outlines the benefits of each function of Enhanced Information Forwarding.

Benefits of Enhanced Information Forwarding

Function	Benefit
Forwarding of original call service requirements (VDN Name or DNIS)	Faster and more efficient agent handling, better service to the caller, and improved CTI integration
Transport of UCID	Improved call tracking as a single call and CTI integration
Collected Digits Transport	Better service to the caller because the caller doesn’t have to repeat input of information, more information for the agent, better and faster call handling, improved call tracking because the collected digits are included with the call record, and improved CTI integration
Forwarding of In-VDN Time	Improved call tracking as a single call and end-to-end time-before-answer statistics
Continued support of ASAI user Information Forwarding	CTI integration
Globally-supported transport	Improved global compatibility

Benefits of Enhanced Information Forwarding (continued)

Function	Benefit
Operation over public networks	improved global compatibility and lower facility/call transport costs
Operation over QSIG trunks	Improved global compatibility and interoperability
Operation over BRI trunks	Lower facility/call transport costs

Network requirements

The network must meet the following conditions for correct Information Forwarding operation:

- Both the private and public networks must support end-to-end transport of codeset 0 user data either as user-to-user information (UUI IE) or QSIG manufacturer specific information (MSI) in the SETUP and DISCONNECT ISDN messages. Private networks can be configured for either non-QSIG transport by way of a codeset 0 UUI IE or QSIG transport by way of MSI packaged in a codeset 0 Facility IE. Currently, public networks do not support QSIG, and user data can only be transported by way of the UUI IE when supported by the network. Future public network offerings may support QSIG. this support may be possibly by way of a Virtual Private Network.
- The switch must support the ISDN country protocol.
- The network byte limit for the user data portion of user information contents must be large enough to carry the data that is needed for the customer application.

Note:

Some public network providers may require service activation and/or fees for user information transport.

Enhanced Information Forwarding has been tested with several major carriers. To find out if these capabilities work with your carrier, check with your account team for the most current information.

If testing has not been done to verify operation over the public networks that are involved with the preferred specific configuration, use of private ISDN trunking between the nodes should be assumed until successful testing is complete.

Enhanced Information Forwarding

The switch offers the following Enhanced Information Forwarding capabilities:

- Forwarding of existing call-related information, including ASAI user data, VDN name, other LAI information such as the in-queue timestamp, and network-provided caller information.
- Forwarding of new call-related information, including collected digits, UCID, and in-VDN time
- Transporting information by way of globally-supported methods
- Providing LAI backward compatibility

For information about administering information transport, see *Administrator Guide for Avaya MultiVantage Software*, 555-233-506. For detailed information about ISDN trunk group setting interactions with Information Forwarding, UCID, and multi-site routing, see Appendix E: Advanced multi-site routing on page 531.

Forwarding of call-related information

Forwarding of call-related information by way of the globally supported transport applies to both BSR and LAI. Depending on administration, information will be sent with an LAI or BSR interflowed call.

Forwarding collected digits with interflowed call

The following summarizes what happens to forwarded collected digits:

- The last set of up to 16 collected digits, not including the dial-ahead digits, are forwarded with a call interflowed over ISDN facilities.
- When processing for the call at the remote location reaches the VDN, the forwarded digits are inserted in the collected digits buffer. Therefore, a TTR is not needed. The objective is to immediately provide the collected digits to the CMS in a DIGITS message and to ASAI by way of the VDN event report in the same manner as incoming ANI.
- The collected digits are available for further routing by steps in the assigned and subsequent vectors, and eventual display to the answering agent.
- All interactions with the collected digits are the same as digits that are collected using a collect step. For example, a subsequent collect step will clear the digits.
- If the call is further interflowed or tandemed over ISDN facilities, the collected digits are tandemed with the call. If more digits are collected at the tandem switch, the latest collected digits are tandemed.

Forwarding accumulated in-VDN time

The following summarizes what happens to forwarded in-VDN time:

- When a call is interflowed, the in-VDN time in seconds, from 0 to 9999, is included. The in-VDN time is the elapsed time starting from the VDN that was originally called until when the Information Forwarding message is created.
- If the call was interflowed to the local system and in-VDN time was received for the call, the previous in-VDN time is added to the local in-VDN time.
- If the accumulated time exceeds the largest value that can be transported, the maximum value is sent.
- The accumulated in-VDN time that is received on an incoming interflowed call is forwarded to the CMS in the DNEVENT message when the call starts VDN/vector processing at the remote location.
- In-VDN time does not pass to the Basic Call Management System (BCMS) for reporting by BCMS.

Transport by way of globally-supported methods

The following summarizes information transport by way of globally-supported methods:

- When a call is LAI or BSR interflowed, the following information is forwarded with the call over public or private ISDN networks using QSIG or non-QSIG protocols:
  - — The LAI information
  - — The collected digits
  - — The in-VDN time data in the ISDN SETUP message.

  The Multi-Site Routing related data is in addition to the associated ASAI user data, which was previously sent in a nonshared UUI IE, and the UCID data.

  Note:

  The forwarded LAI information is the same as what was previously sent in the LAI IE: VDN name also called LAI DNIS, put in queue time-stamp, priority level and type of interflow.

- The other call related information, including calling party number (ANI), calling party name, II-digits and CINFO digits, that is tandemed with the interflowed call in the SETUP message is forwarded in the same manner as prior to Release 6.

  Note:

  II-digits and CINFO are forwarded as codeset 6 IEs which may be a problem in some switched networks.

- At the remote end, the transported data is separated into its component parts for storage with the call, call vectoring, call processing and display, further interflow or tandeming, and forwarding to adjuncts. For example, the LAI info is treated as though it was received as an incoming codeset 6 LAI IE including forwarding over ASAI as a code set 6 LAI IE in event reports.
- When a status poll call is placed to the remote location, the switch only forwards the UCID and caller information that was received from the original call.
- In response to a status poll, the switch forwards the reply-best status data in the ISDN DISCONNECT message over public or private ISDN PRI/BRI networks. In this case, the DISCONNECT message has a cause value of 31 “Normal–Unspecified” for wider international interoperability.

Providing LAI backward compatibility

The following summarizes backward compatibility issues:

- There is a trunk group option to specify whether to include an LAI IE (codeset 6 or 7) in the SETUP message for LAI interflowed calls. When this option remains as the default **y**, an LAI interflow (using the existing or enhanced LAI vector command) will include a codeset 6/7 LAI IE per existing operation for inter-operability with a mixture of old and new Avaya switches. The option must be **n** if the network does not support codeset 6/7 and/or this IE is not required. With trunk groups to all of the R6.3 and newer switches, this option should always be **n**.

  Note:

  This option cannot be used with BSR calls because the remote switch must be an R6.3 or newer switch in order to work with BSR and codeset 0 information transport by way of shared UUI is required for BSR polling calls.

- Administer the ISDN Trunk Group option: `Send Codeset 6/7 LAI IE`. This option is valid even if LAI at the remote site is not active for tandem situations. Use of this option for LAI does not depend on the setting of the Vectoring Best Service Routing customer option.
- If the ISDN trunk group option is set to send the LAI IE, this IE is sent in addition to the Information Forwarding by way of codeset 0 shared UUI transport when a call is LAI interflowed over a trunk in this trunk group. With shared UUI, you can set the LAI data to not be included in the UUI IE.
- Administer the Shared UUI priorities. This is important when the network byte limit on the user data part of the UUI IE user information contents is not large enough to carry the data that is needed for the customer application. Note that Shared UUI priorities do not apply to QSIG. To determine customer application data sizes, see Determining user information needs. For instructions on how to administer Shared UUI, see *Administrator Guide for Avaya MultiVantage Software*, 555-233-506.

Determining user information needs

The network byte limit on the user data part of the UUI IE user information contents must be large enough to carry the data that is needed for the customer application.

Note:

The UUI IE uses 3 bytes for the header information and allows from 32 bytes to 128 bytes for the user data portion. For example, if the network specifies that it can transport 32 bytes of user data, the UUI IE length is 35 bytes.

The user information capacity need is determined by adding the space that is required for each data item to be transported based on the following rules:

- Each included shared data item requires 2 bytes for the header plus the data.
- If the data item priority is set to blank in the Shared UUI Feature Priorities form, the data item is not sent and no space is allocated for it.
- For most data items, the data byte length depends on the configuration of the customer application except for UCID, In-VDN time, and Other LAI. These applications have a fixed byte length.
- If the administered Maximum UUI IE Size is exceeded, the lowest priority items are not included until the remaining data fit. If a specific data item at a higher priority exceeds the administered UUI IE size setting, that item is not sent, leaving room for other lower priority items.
- Only a maximum of 128 bytes of user data is supported by the switch with UUI. Non-QSIG private networks support the full capacity. Non-QSIG public networks may support less than the full 128 bytes, for example, 32 bytes.
- QSIG signaling and networks do not have user information size limits. They will support sending MSI for user data items at their maximums. Determination of space allocation and administration of priorities does not need to be done for QSIG networks.
- Since ASAI user data can be up to 96 bytes (98 bytes with the header), the need for other interflow shared data transport must be carefully considered in setting priorities and determining how much ASAI user data to support for the application. If the network supports the full 128 bytes and all interflow data at their maximums is to be transported (48 bytes), a total of 78 bytes of ASAI user data (80 bytes with header) is allowed. If the full 96 bytes of ASAI user data is required (98 bytes with the header), then only 30 bytes is available for other interflow data.

Note:

If the network supports 128 bytes and 78 bytes or less of ASAI user data is required, you do not need to determine space allocation or administer priorities.

Use the following table to determine the space that is required to send the various user data for your applications.

Bytes required to send user data

Type of user data	Total user data bytes (with 2-byte header)	This type of user data...
ASAI	2 to 98 or 0 (calculated by 1 byte per byte of ASAI user information)	is only needed with certain CTI applications. Space is only required when the CTI application sends user information and the amount of space is determined by the application. For example, 34 bytes is required if the application sends 32 bytes of data. Sending more than 78 bytes of ASAI data (80 bytes with the header) reduces capacity for other interflow data.
UCID	10 or 0	works with BSR to track calls across multiple sites. This data item may not be included even if the priority is set to “1” depending on the trunk group setting and/or system feature settings. When the data item is not included, it does not take up any space.
In-VDN Time	4	works with BSR to determine time before answer for calls and tracking as a single call across sites. It could be eliminated for calls with a short waiting time. If the priority field is not blank, it is always included.
VDN Name	2 to 17 (calculated by 1 byte per character in name (max 15))	works with BSR but could be eliminated if dedicated VDNs are used at receiving sites with names assigned that display the equivalent information to the answering agent. An interflowed call that is received without the originating VDN name uses the incoming VDN name. If the priority field is not blank, the 2-byte header is always included.
Collected Digits	4 to 11 or 0 (calculated by 1 byte per 2 digits plus 1 (max 16 digits))	requires a whole byte for an odd number of digits. For example, 1 digit needs 2 bytes (1 plus 1), 7 digits need 5 bytes (4 plus 1), and 16 digits need 9 bytes (8 plus 1).
Other LAI Info	6	is needed for existing CTI applications that require this data, including in-queue time stamp, queue priority, and interflow type, from the LAI IE.

Example

Assume that your public network supports only 32 bytes of user information. Your application requires 13 bytes of ASAI user information (15 bytes of user data), UCID (10 bytes of user data), and 8 collected digits (7 bytes of user data, that is, 4 plus 1 plus 2 for the header). It does not require Other LAI Information. Also, calls spend very little time at the sending switch because the calls are not queued before interflow takes place and tracking as a single call is not required.

By dedicating appropriately named VDNs at the receiving switch, the public network can support the application. Because the needed data items require the entire 32 bytes of user data, the priority fields for the `In-VDN Time`, `VDN Name`, and `Other LAI Information` must be set to blank.

Simple troubleshooting for Information Forwarding

The following troubleshooting hints should be reviewed when information is not forwarded, even though you received no error messages while administering the Shared UUI feature, and all software and connections meet the minimum requirements:

- If DCS is used, ensure that all ISDN trunks between the switches that are used for DCS or remote AUDIX are configured in the D-channel mode.
- For each ISDN trunk that is administered with the Shared UUI option, make sure that the UUI size does not exceed the UUI IE size that the network can support. For more information, see Determining user information needs on page 158.
- Make sure that trunk group options are set correctly for the application and configuration.
- Applications may fail on networks supporting limited UUI transport. Administration determines which application's UUI will be transported in these cases. If a given application is failing, first check the administration to determine if the application in question has the highest priority. This applies to tandem nodes as well as to originating nodes.
- Applications that originate UUI on tandem nodes can request that assigned priorities at the tandem node be applied to the resulting UUI. Therefore, it is possible for a tandem node to erase UUI information that was received from the originator. Passing UUI through a tandem node transparently as required for UUS Service 1 does not apply to the switch proprietary shared UUI procedures.
- When a new application is implemented, run the “display events” command on a periodic basis for the appropriate vector. The resulting report notifies you if any enhanced information could not be sent.

Chapter 9: Adjunct (ASAI) Routing

Adjunct Routing provides a means for an adjunct-switch application interface (ASAI) processor to specify the destination of a call when it encounters an **`adjunct routing`** vector command during vector processing.

An adjunct is any processor that is connected to a switch that can use the ASAI protocol. The adjunct makes a routing decision according to caller information and/or agent availability, and returns a call route response to the switch.

The switch provides information in an ASAI route request message that the adjunct application uses to access a database and determine a route for the call. In a typical application, the ASAI adjunct might use the dialed number, the Calling Party Number (CPN/BN), or the digits that are collected by way of Call Prompting to access caller information and thereby determine an appropriate call route.

Adjunct Routing can be used in conjunction with the Call Prompting and Look-Ahead Interflow features. When combined with one of those features, the following rules apply:

- When combined with Call Prompting, Adjunct Routing can pass up to 16 digits that are collected from the last relevant **`collect digits`** vector command.
- When combined with Look-Ahead Interflow (LAI), Adjunct Routing can pass the LAI information element or other call center-related data (with enhanced Information Forwarding) that was passed from the originating switch in the ISDN message or associated with the call from the local switch.

This chapter includes the following major topics:

Considerations for implementing adjunct routing

You should understand the following considerations before you implement a call center solution that uses the Adjunct Routing feature:

- An adjunct specified in an **`adjunct routing`** command can route a call to an internal number, an external number, a split, a VDN, an announcement extension, or a particular agent. An adjunct can also provide priority ringing, priority queuing, and specify that a route-to an agent be done as a direct agent call.
- If your specific application permits you to do so, you can include two or more consecutive **`adjunct routing`** steps in a vector. This approach provides the following advantages:
  - — Redundancy in case of ASAI link/application failure.
  - — Simultaneous processing of multiple route requests, which distributes incoming call load more efficiently and results in faster call processing times. For more information, see Multiple outstanding route requests on page 179.
- Vector processing continues to occur while an ASAI route request is being processed. For this reason, the first step to follow one or more adjunct routing steps should be either an **`announcement`**, or a **`wait time`** step that adheres to the following rules:
  - — If an **`announcement`** step follows immediately after an **`adjunct routing`** step, the announcement should not contain any information that is essential to the caller (such as further instructions), since it will immediately terminate when the switch receives a destination from the ASAI adjunct.
  - — If a **`wait-time`** step follows immediately after an **`adjunct routing`** step, it should usually specify either **`ringback`** or **`music`** (but not **`silence`**) as the feedback option, so that the caller is less likely to abandon the call.

Important:

If an ASAI link/application specified in the **`adjunct routing`** step is out of service, the step is skipped. If the next step is not a **`wait-time`**, **`announcement`**, or **`adjunct routing`** step, as much as six minutes may elapse before the switch determines that the adjunct application is out of service.

- The second step after the **adjunct routing** step can, and often should, be implemented as a default treatment in case the host application or ASAI link is down. Speed of execution for the default treatment step (for example, **route-to number 0 if unconditionally**) is controlled by the following factors:
  - If the ASAI link is down, and if the first non-adjunct routing step is either a **wait-time** or an **announcement** treatment, then the treatment step is skipped and the default step that follows the skipped treatment executes immediately.
  - If the host application is not down, the default step executes only if the adjunct does not provide a route within the time defined by the first non-adjunct step. For example, if the first non-adjunct step is an announcement, the default step executes only after the time defined by the length of the announcement is exceeded.
- When a vector contains an **adjunct routing** command, and an ASAI link/application failure event occurs, special rules apply to vector processing operations that result. Adjunct Routing vectors should be designed to take these special processing operations into account. For more information, see Special vector processing considerations associated with adjunct routing on page 169.
- Since vector processing continues to occur while an ASAI call route request is processed at an adjunct, succeeding vector steps can terminate an ASAI call route request if they execute before a call route can be provided by the adjunct. Alternately, the adjunct may reject the call route request, and subsequent vector processing proceeds in a normal manner. For more information, see Vector steps that terminate an ASAI call route request on page 174.
- The **wait-time hearing i-silent** command is used in cases where it is important to allow the adjunct to decide whether to accept an incoming ISDN-PRI call. When this step is encountered after an **adjunct routing** step, the switch does not return an ISDN PROGress message to the originating switch. This is particularly important for Network ISDN features and the Look-Ahead Interflow feature.

Receiving and implementing an ASAI call route

A switch that receives an adjunct-supplied call route performs various checks to validate the call route before it is implemented. When the adjunct-supplied route is validated, the operations that result are similar to those in effect for a **`route-to`** ***`xxxxx`*** **`with coverage=y`** command. The caller hears normal call progress tones and feedback, and if the call routes to an extension with no available call appearances and no coverage path, the caller hears a busy signal.

Any other features that may be in effect at the adjunct-supplied destination, such as Send-All-Calls or Call Forwarding, interact with the routed call.

Also, Look-Ahead Interflow operations are not applied when calls are routed over ISDN trunks. Instead, ASAI-routed calls are directed to their adjunct-supplied destination without waiting for call acceptance.

The processes associated with receiving and implementing and ASAI call route are described in the following sections:

- Validation requirements for an adjunct-supplied call route on page 166
- Switch response to validated adjunct-supplied call routes on page 167
- Switch response to invalid adjunct-supplied call routes on page 167

Validation requirements for an adjunct-supplied call route

When the switch receives adjunct-supplied call route instructions, the switch validates the route according to the following process:

1. The switch verifies that the COR rules specified for the target VDN permit the call to be terminated at the adjunct-supplied destination.
2. The switch validates the following information:
   - Destination number
   - ACD split
   - TAC/AAR/ARS access code
   - Dial plan compatibility
   - Other options specified by the adjunct
3. If the ASAI adjunct specifies the Direct Agent Call (DAC) option, the destination number (agent) must be logged into the adjunct-specified ACD split.
4. If the destination for the call is external, the switch verifies that a trunk is available for the call.

Switch response to validated adjunct-supplied call routes

If the switch validates an adjunct-supplied call route, the following operations occur:

1. Vector processing in the VDN that contains the initiating **`adjunct routing`** command terminates immediately.
2. The switch signals the ASAI adjunct that the route is accepted.
3. The switch routes the call to the destination specified by the ASAI adjunct.

Switch response to invalid adjunct-supplied call routes

If any of requirements for call route validation listed in Validation requirements for an adjunct-supplied call route on page 166 are not met, items the following operations occur:

1. The switch discards the route.
2. The switch signals the ASAI adjunct that the route is invalid.
3. Vector processing of any other default treatment steps in the VDN that contains the initiating **`adjunct routing`** proceeds.

Data sent with an ASAI call route request

When a call encounters an **`adjunct routing`** command and if the call is not queued to a split, the switch sends an ASAI message that requests a call route over the specified adjunct link. The following list identifies the contents of the message, along with a comment or a brief explanation for each item:

- **Calling number information**. The calling party number or billing number (CPN/BN) that is provided by ISDN-PRI or R2MFC signaling facilities. If the call originates from a local switch extension, this extension is the calling number.
- **Originating line information (II-digits)**. A two-digit code that is provided by ISDN-PRI facilities that indicates the type of originating line.
- **Called number**. The originally called extension if a call is forwarded to a VDN, or the first VDN through which the call was routed if the call was not forwarded to the VDN.
- **Routing VDN**. The last VDN that routed the call to the vector that contains the **`adjunct routing`** command.
- **Call identifier**. An ASAI identifier that permits the ASAI adjunct to track multiple calls by either Event Notification or 3rd Party Call Control. For more information on ASAI, see *Avaya MultiVantage CallVisor ASAI Technical Reference*, 555-230-220.
- **Enhanced Information Forwarding (related data) and Look-Ahead Interflow information (if any).** Includes the original VDN display information, the priority level of the call at the originating switch, and the time that the call entered vector processing. For more information, see Look-Ahead Interflow (LAI) on page 203, and Information Forwarding on page 151.
- **Digits collected by Call Prompting or Caller Information Forwarding (CINFO)** (if any; maximum of 16 digits). Digits that are collected by the most recent **`collect digits`** command. For more information, see Call Prompting on page 181, ANI /II-digits routing and Caller Information Forwarding (CINFO) on page 139, and Information Forwarding on page 151.
- **User-to-User Information** (UUI). User-provided data that is associated with the call. If provided by ASAI, this data was provided in a 3rd-Party-Make-Call, Auto-Dial, or Route-Select message. If provided over ISDN, the data was in the SETUP message that delivered the call to this switch. Calls that contain UUI specifically used by ASAI allow ASAI UUI to be propagated to the new call during a manual transfer or conference operation. ASAI UUI is propagated to a new call during its establishment when the agent presses the transfer/conference button the first time. If the call is transferred to a remote switch, the ASAI UUI from the first call is copied into the SETUP message sent for the second call, in which case, the alerting event message sent to an ASAI application contains the ASAI information.

Special vector processing considerations associated with adjunct routing

When you design call vectors that include one or more **adjunct routing** commands, you must be aware of a number of special operational features. These considerations are described in the following sections:

- Effects of ASAI link/application failure on vector processing on page 169
- Simultaneous processing of vector steps and ASAI call route requests on page 173

Effects of ASAI link/application failure on vector processing

An ASAI link failure can change the manner in which subsequent **announcement** or **wait-time** treatment steps are processed.

In the following simplified vector example, the step that follows immediately after an **adjunct routing** command is a **wait-time** command. If the adjunct routing step fails at either the ASAI link or adjunct application, the wait-time step is skipped.

The second step after the **adjunct routing** step is often implemented as a default treatment. In the example shown above, the default treatment in step 3 is a route to an attendant. If the switch recognizes that the ASAI link or adjunct application is out of service, this step executes immediately. Otherwise, the step executes only if the application does not respond with a route within 60 seconds (the wait-time assigned in the example).

Simplified example of vector processing in an ASAI link/application failure condition

```
1. adjunct routing link 11 . . . . . . . . . . . . . . (link/application is down)
2. wait-time 60 seconds hearing ringback . . . . . . . (step is skipped)
3. route-to number 0 with cov n if unconditionally . . (step is executed)
4. disconnect after announcement 2000
```

Vector processing with goto steps in an ASAI link/application failure condition

Processing rules for a vector that includes one or more **`adjunct routing`** commands and has an ASAI link/application failure condition in effect are summarized as follows:

- An **`announcement`** or **`wait time`** treatment is skipped whenever one of the following conditions is true:
  - — The treatment step follows immediately after a failed **`adjunct routing`** command
  - — The treatment step is the first non-goto step that follows a **`goto`** step that succeeds. In this context, a **`goto`** step is considered to succeed when the specified **`goto`** condition is true, and the call branches from the **`goto`** step to the treatment step.
  - — The treatment step is the first non-goto step that follows a failed **`goto`** step. In this context, a **`goto`** step is considered to fail when the specified **`goto`** condition is true, the call fails to branch, and control proceeds to the treatment because it is the next step listed in the vector sequence.

Note:

The treatment step is skipped even when a failed **`goto`** step that precedes it is, in turn, preceded by one or more successful **`goto`** steps.

The rules listed above for vector processing under ASAI link/application failure conditions are further illustrated in the following examples.

Example 1 - Vector processing with goto steps in an ASAI link/application failure condition

```
VDN (extension=1040  name=‘‘Ad Route’’  vector=40)
Vector 40

 1. adjunct routing link 10 . . . . . . . . . . . . . . . (link/application is down)
 2. wait-time 10 seconds hearing ringback . . . . . . . . (step is skipped)
 3. adjunct routing link 20 . . . . . . . . . . . . . . . (link/application is down)
 4. goto step 7 if available-agents in split 20 < 1 . . (step executes and condition is false)
 5. wait-time 10 seconds hearing ringback . . . . . . . . (step is skipped)
 6. goto vector 50 if unconditionally . . . . . . . . . . (step executes, go to vector 50)
 7. goto step 10 if calls-queued in split 20 pri l > 50
 8. announcement 4001
 9. goto vector 50 if unconditionally
10. route-to number 6000 with cov n if unconditionally
    VDN (extension=6000  name=‘‘Message’’  vector=60)
```

Based on the scenario presented in the example shown above, the following vector processing events occur:

Step 1 fails – For purposes of this example, assume that the adjunct link or application is out of service. The **`adjunct routing`** command in step 1 fails.

Step 2 is skipped – Because the **`wait-time`** command in step 2 immediately follows an **`adjunct routing`** command whose adjunct link is out of service, the **`wait-time`** step is skipped.

Step 3 fails – For purposes of this example, step 3 contains another **`adjunct routing`** command whose adjunct link is assumed to be out of service. The step fails, and control is passed to the **`goto step`** command in step 4.

Step 4 executes – A **`goto`** step that immediately follows a failed adjunct routing command is always executed. In this example, the command fails to branch because there is at least one available agent in split 20.

Step 5 is skipped – The **`wait-time`** step that follows the unsuccessful **`goto`** step (step 4) is skipped, because in an ASAI link failure condition, the first non-goto step to be processed after the first successful first goto step is always skipped if it is either **`announcement`** or **`wait-time`**. Control is passed to the **`goto vector`** command in step 6.

Step 6 executes – Step 6 routes the call to vector 50 (not shown), which is designed to queue the call and provide standard call treatment.

In the next example, assume that the **`goto step`** command in step 4 succeeds. In this context, the **`goto`** step succeeds when the specified condition is true (no agents are available in Split 20), and control is passed to step 7, where another **`goto`** step determines whether there are more than 50 calls in split 20. If the condition is true, step 7 succeeds and control is sent to step 10, where the **`route-to number`** command sends the call to vector 60.

The example processing events are described in detail below.

Example 2 - Vector processing with goto steps in an ASAI link/application failure condition

```
VDN (extension=1040  name=''Ad Route''  vector=40)
Vector 40

 1.  adjunct routing link 10 . . . . . . . . . . . . . . . (link/application is down)
 2.  wait-time 10 seconds hearing ringback . . . . . . . (step is skipped)
 3.  adjunct routing link 20 . . . . . . . . . . . . . . . (link/application is down)
 4.  goto step 7 if available-agents in split 20 < 1 . . (step executes and condition is true)
 5.  wait-time 10 seconds hearing ringback
 6.  goto vector 50 if unconditionally
 7.  goto step 10 if calls-queued in split 20 pri l > 50 (step executes and condition is true)
 8.  announcement 4001
 9.  goto vector 50 if unconditionally
10.  route-to number 6000 with cov n if unconditionally  (step executes unconditionally)


VDN (extension=6000  name=''Message''  vector=60)
Vector 60

1.   announcement 4000 (''We're sorry.  We are still . . (step executes)
     unable to connect you to an agent. If you'd like to
     leave a message, please do so after the tone.
     Otherwise, please call back weekdays between 8:00
     A.M. and 5:00 P.M. Thank you.'')
2.   wait-time 6 seconds hearing silence
3.   messaging split 18 for extension 1500
4.   announcement 4010 (''We're sorry.  We were unable to
     connect you to our voice mail. If you'd like to try
     to leave a message again, please do so after the
     tone.  Otherwise, please call back weekdays between
     8:00 A.M. and 5:00 P.M. Thank you.'')
5.   goto step 2 if unconditionally
```

Based on the scenario presented in the example shown above, the following vector processing events occur:

Step 1 fails – For purposes of this example, the adjunct link or application is out of service. The **`adjunct routing`** command in step 1 fails.

Step 2 is skipped – Because the **`wait-time`** command in step 2 immediately follows an **`adjunct routing`** command whose adjunct link is out of service, the **`wait-time`** step is skipped.

Step 3 fails – For purposes of this example, step 3 contains another **`adjunct routing`** command whose adjunct link or application is also out of service. The step fails, and control is passed to the **`goto step`** command in step 4.

Step 4 executes – A **goto** step that follows a failed adjunct routing command is always executed. In this example, the command succeeds and branches to step 7, because no agents are available in split 20.

Step 7 executes – Again, a **goto** step that follows a failed adjunct routing command is always executed. In this example, the command branches unconditionally to Vector 60

Step 10 executes – In this example, step 10 (**route-to number**) is the first non-goto step immediately preceded by one or more **goto** steps in an ASAI link fail condition. The step executes, because it not an **announcement** or **wait time** command.

Vector 60: Step 1 executes – The first step in this vector is an **announcement** command. In this example, this is the first step in the processing sequence to be either an announcement or wait time step. However, this step is not skipped, since it is not the first non-go to step in the processing sequence. Instead, step 10 in Vector 40 (a **route-to number** step) is the first non-goto step.

Simultaneous processing of vector steps and ASAI call route requests

When the switch sends a route request to an ASAI adjunct, vector processing continues for any vector steps that follow the **adjunct routing** command. Therefore, non-adjunct routing step that follows immediately after an **adjunct routing** step (or multiple adjunct routing steps in uninterrupted succession) can determine:

- The maximum length of time that the switch waits for a call route reply from the ASAI adjunct
- In some cases, whether or not the ASAI call route request is allowed to finish processing

If the next step is not a **wait-time**, **announcement**, or another **adjunct routing** command, as much as six minutes may elapse before the switch determines that the adjunct application is out of service. For this reason, the recommended practice is to design vectors so that the next step to follow an **adjunct routing** command is either a **wait-time**, or **announcement** step.

Vector steps that terminate an ASAI call route request

If an **adjunct routing** step is followed by a **wait-time** or **annoucement** treatment, and the treatment completes before an ASAI call route request is returned by the adjunct, call processing continues for any vector steps that may follow the treatment. In this case, certain vector commands will terminate the ASAI call route request when they are executed. Vector commands that terminate an active ASAI call route request include:

- **busy**
- **check split**
- **converse-on split**
- **queue-to split**
- **collect digits**
- **disconnect**
- **messaging split**
- **route-to**

If a valid ASAI call route message is received by the switch before one of the vector commands listed above can execute, the system routes the call to the destination specified by the adjunct route. Otherwise, the ASAI route request is terminated.

Note:

The adjunct can also reject a call request by “negatively acknowledging” the route request that is sent by the switch. When the switch receives a a route request rejection message from the adjunct, any **announcement** or **wait-time** step that is being executed is immediately terminated. Call processing then continues with the next vector step.

Adjunct routing-initiated path-replacement

Path replacement for calls in queue and vector processing, using QSIG or DCS with Reroute using ISDN SSE, is available for Avaya switch software R9.5 or later. For calls that are waiting in queue or in vector processing, even if the call is not connected to an answering user, path replacement can be attempted to find a more optimal path for this call. This results in more efficient use of the trunk facilities.

When adjunct routing is used with a call, path-replacement can be initiated when the following criteria are true:

- The inbound call is over an ISDN QSIG trunk or ISDN DCS SSE trunk
- A route-select response is received from the CTI application after the **`adjunct route`** vector command has been executed
- The routing destination that is contained in the route select ASAI message is to an outbound ISDN QSIG trunk or out bound ISDN DCS SSE trunk

When all three criteria are met, the trunk is then seized and used for the call.

Note:

This ability to track a measured ACD call after path replacement has taken place is available for CMS versions r3v9ah or later.

Example vector for adjunct routing-implemented path replacement

The following Call Vector example shows how a vector for adjunct routing can be written to trigger path-replacement at the terminating switch.

Note:

In order for a path-replacement to be attempted, the incoming and outgoing trunks that are used for the call must be administered with the `Supplementary Service Protocol` field set to b.

Adjunct routing-initiated path-replacement vector

```
1.   announcement 5996
2.   adjunct routing link 3454
3.   wait 20 seconds hearing ringback
4.   announcement 3111
```

At the terminating (receiving) switch, the vector that is executed by the incoming call must be programmed with an **`announcement`**, **`wait hearing music`**, or **`wait hearing ringback`** vector command. The use of one of these commands is what makes it possible for path-replacement to take place while the call is in vector processing.

Phantom calls

A phantom call is a call that originates from a nonphysical device by way of an ASAI application and may be placed anywhere. In general, phantom calls

- Use less resources
- Are treated like voice calls

How do phantom calls work?

First, an application requests a phantom call by sending an ASAI `third_party_make_call` or `auto_dial` capability message to the switch.

If the specific extension of a station administered without hardware (AWOH) is specified as the originator, the switch places the call from that extension if the extension is available.

It is also possible to specify a hunt group extension with members that are AWOH extensions as the originator.

How are phantom calls used?

Applications use phantom calls when they need to originate a call without using a physical device and thus not use extra resources. For example, applications may need to:

Reserve a queue slot – Many contact centers handle incoming requests as voice, video, data, voice messages, faxes, and e-mail. Agents who work in these contact centers need to handle the mix of requests. However, a single queue needs to manage and distribute the work load for these agents.

For each non-voice request, the application can place a phantom call into the queue. When the phantom call reaches the head of the queue, it is delivered to the agent. The agent is then given the corresponding work item on the desktop, for example, the fax.

Conference control – Multiple parties (both internal and external) can be conferenced into a call. The initial call is placed as a phantom call. When answered, the call is placed on hold by the application and another phantom call is made. The two calls are then conferenced together. This process is repeated until all parties are added to the call.

Help with trunk-to-trunk transfers. – Working with the Single Step Conference feature, applications can use the phantom call feature to help with trunk-to-trunk transfers, that is, transferring a trunk-to-trunk call to another trunk. For information about single step conferences, see the *Avaya MultiVantage CallVisor ASAI Technical Reference*, 555-230-220.

Alerts (wake-up, maintenance, and security) – Applications can use phantom calls to alert users of various conditions such as wake-up, maintenance, or security.

How do phantom calls affect Call Vectoring?

Because phantom calls can be directed anywhere, you must properly configure the application and the switch to ensure that the vector commands that are executed for these calls make sense. For more information, see the *Avaya MultiVantage CallVisor ASAI Technical Reference*, 555-230-220.

The switch does not block phantom calls from executing any vector commands because phantom calls follow the same vector processing as regular voice calls. However, it might not make sense to have phantom calls enter certain vector steps such as:

Announcements – Because there is nobody listening to an announcement that is made to a phantom call, there is no sense in playing one.

"collect" steps – In a phantom call, the **`collect`** step fails because it can not connect a tone receiver to a station AWOH (Administration without hardware); it times out because there is nobody to put in the expected digits.

The **`busy`** step provides a busy signal to the caller. In a phantom call, the **`busy`** step disconnects the call because the switch clears a phantom call when the call cannot terminate at a specific local destination.

Phantom call administration

There are no administration forms that are specific to phantom calls, but the following criteria must be met in order for the feature to work:

- Some stations AWOH must be administered.
- If a hunt group is specified as originator, a non-ACD hunt group with AWOH members must also be administered.
- It is recommended that meaningful names are assigned for the stations AWOH that are used by phantom calls if the calling party name will appear on the agent's or Service Observer's display.

Single-step conference

The Single-Step Conference (SSC) feature is available for Avaya switch software R6.3 or later. SSC allows an application to:

- Add a device into an existing call, for example, to play announcements or make voice recordings
- Facilitate application-initiated transfers and conferences

Stations that are AWOH are eligible for single-step conference. The party may be added to a call in listen only mode (no visibility) or with listen and talk capability (visibility).

Single-step conference is only available through an ASAI link. For more information about single-step conference, see the *Avaya MultiVantage CallVisor ASAI Technical Reference*, 555-230-220.

How does SSC work with Call Vectoring?

The call to which an extension is to be single-step conferenced is not allowed to be in vector processing unless the visibility option with the single-step conference request indicates “no visibility.”

Multiple outstanding route requests

This feature allows multiple ASAI route requests for the same call to be simultaneously active. The route requests can be over the same or over different ASAI links.

Route requests are all made from the same vector. They must be specified without intermediate (**`wait-time`**, **`announcement`**, **`goto`**, or **`stop`**) steps. If the adjunct routing commands are not specified back-to-back, standard adjunct routing functionality applies and previous outstanding route requests are cancelled when an adjunct routing vector step is executed.

The first route select response that is received by the switch is used as the route for the call and all other outstanding route requests for the call are canceled.

With multiple outstanding route requests, multiple adjuncts can process the route call request without waiting for the first route attempt to fail. An application can make use of this feature to distribute the incoming call load evenly across adjuncts based on the adjunct's current CPU load.

Note:

Each link has a unique extension number, even in a configuration where there might be multiple links to the same adjunct.

Multiple call route request example

The following example shows a typical vector where multiple adjunct route requests to multiple links are active at the same time. The first adjunct to route the call is the active adjunct and it specifies which VDN the call should be routed to at that point.

Sample adjunct routing vector with redundancy

```
1.   wait-time 0 seconds hearing ringback
2.   adjunct routing link 1001
3.   adjunct routing link 1002
4.   adjunct routing link 1003
5.   wait-time 6 seconds hearing ringback
6.   route-to number 1847 with cov n if unconditionally
     default routing)
```

Chapter 10: Call Prompting

Call Prompting provides flexible call handling that is based on information that is collected from a calling party. This information is in the form of dialed digits that originate from an internal or external touch-tone telephone or from an internal rotary telephone that is on the same switch as the vector. Call Prompting allows for the temporary transfer of call management control to the caller.

With Call Prompting and Vectoring enabled, the switch can collect caller entered digits (ced) and customer database provided digits (cdpd) that are supplied by the network. The system can receive Call Information Forwarding (CINFO) digits in an incoming call's ISDN message when the AT&T Network Intelligent Call Processing (ICP) service is in use. A switch can collect digits and forward those digits to other switches by way of interflow commands. For more information, see Caller Information Forwarding on page 147.

With Voice Response Integration (VRI), digits can be returned to the switch by a Voice Response Unit (VRU) script that is accessed by a **`converse-on split`** command. Such digits can also be used for call management.

Call Prompting can be used in various applications so that calls can be handled with more flexibility.

This chapter includes the following topics:

Command set

The following table show the commands that are used for Call Prompting.

Call Prompting command set

Command category	Action taken	Command
Information collection	Collect information from the calling party, from the public network in an ISDN SETUP message, from a Voice Response Unit (VRU), or from CallVisor ASAI.	**collect digits**
Treatment	Play an announcement. Delay with audible feedback of silence, ringback, system music, or an alternate audio/music source.	**announcement** **wait-time**
Routing	Leave a message. Route the call to a number that is programmed in the vector. Route the call to digits that are supplied by the calling party.	**messaging split** **route-to number** **route-to digits**
Branching/ programming	Go to a vector step. Go to another vector. Stop vector processing.	**goto step** **goto vector** **stop**

Touch-tone collection requirements

Before the switch can accept the touch-tone digits that are entered by a caller, the switch must be equipped with a “collection resource.” The resource used for collecting and interpreting touch-tone digits is a unit of hardware called a Touch-Tone Receiver (TTR). These TTRs are provided on the call classifier and tone detector circuit packs, one of which is required for Call Prompting.

The number of TTRs that are required is configured according to two sources:

- Customer input to the Avaya Account Team
- Account team input to the configurator tool

For existing systems that are adding a Call Prompting application, the Account Team recommends the appropriate number of TTRs based on two factors:

- Account team input to the configurator tool
- Application review by the Avaya Design Center

The process of collecting CINFO digits does not require TTRs.

Outside callers must have a touch-tone telephone to enter the digits that are requested by the **`collect digits`** command. For callers who are using rotary dialing, the Call Prompting timeout takes effect, the **`collect digits`** command times out, and vector processing continues at the next step. As a precaution, always provide a default treatment, such as a **`route-to attendant`** command or a **`queue-to split`** command, in the vector script unless the script is created exclusively for users of touch-tone telephones.

Note:

The Call Prompting interdigit timeout can be administered for any number of seconds from 4 to 10. This value is administered on the Feature-Related System Parameters form.

Provisions for users of rotary telephones are illustrated in the vector scripts in this chapter.

Call Prompting digit entry — collect digits command

The touch-tone digits that are entered by a Call Prompting user are collected by the **`collect digits`** command. This command allows the system to collect up to 24 digits from a touch-tone telephone. Sixteen of these digits may be collected immediately, while any remaining digits are stored as dial-ahead digits, which are explained later in this chapter.

Call Prompting allows some flexibility in entering digits. Specifically, the caller can:

- Remove incorrect digits strings
- Enter variable-length digit strings
- Enter dial-ahead digits.

The following sections explain these processes.

Removing incorrect digit strings

An announcement that requests the caller to enter digits can be included in call treatment. As an option, the announcement can instruct the caller to enter an asterisk (*) if he or she enters incorrect data.

When the caller enters a “*”, the following happens:

1. Digits that were collected for the current **`collect digits`** command are deleted.

   Note:

   Also deleted are any dial-ahead digits that are entered and that do not exceed the maximum digit count of 24. (Dial-ahead digits are explained later in this chapter.)

2. Digit collection is restarted.
3. The announcement is not replayed.

Once the caller enters an asterisk, the caller can reenter digits for processing.

Entering variable-length digit strings

The maximum number of digits that are requested from the caller must be specified in the administration of the **`collect digits`** command. In some cases, the caller might be permitted to enter fewer digits than the maximum specified. In fact, the number of digits that the caller enters can vary for several variations of one **`collect digits`** command. Each such grouping of digits is called a variable-length digit string.

Call Prompting allows for variable-length digit strings by providing an end-of-dialing indicator in the form of the pound sign (#). The pound sign is used to end any digit string that is entered by the caller, and it does the following:

- Tells the system that the caller has finished entering digits
- Causes the next vector step to be processed immediately.

Whenever the caller is permitted to enter a variable-length digit string, the announcement portion of the **`collect digits`** command should specify the largest possible number of digits that can be entered. Accordingly, each **`collect digits`** command should be administered to collect no more than the intended maximum number of digits. The caller can enter a pound sign part of a variable digit string entry either:

- At the end of each variable digit string that is entered. In this case, the pound sign should be included in the count of the number of maximum digits that can be entered.
- At the end of each such string that, not counting the pound sign, contains fewer characters than the maximum number of allowable digits. In this case, the pound sign should not be included in the count of the number of maximum digits that can be entered.

If the caller enters more digits than the maximum number specified, the additional digits are saved as dial-ahead digits for subsequent **`collect digits`** commands. If the vector or vectors chained to it do not contain another **`collect digits`** command, the extra digits are discarded.

If the caller enters fewer digits than the maximum number specified and does not complete the entry with the pound sign, a Call Prompting timeout occurs. The timeout terminates the command, and any digits collected prior to the timeout are available for subsequent vector processing.

A common application involving the entering of variable-length digit strings allows the user to dial either the number for the attendant or an extension to reach the desired destination. If the maximum number of digits that can be entered is administered to be 3 and the user wishes to reach the attendant, the user should dial “0#.” However, if the user chooses to dial a 3-digit extension, the user should dial, for example, “748” and not “748#.” Since the maximum number of digits that can be dialed in this case is three, dialing “748#” would cause “#” to be saved as a dial-ahead digit. On the other hand, if the caller dials “748#,” and if the maximum number of digits that can be entered is 4, “#” is not saved as a dial-ahead digit since it is the fourth of four digits that can be entered in this case.

Entering dial-ahead digits

When digit collection for the current **`collect digits`** command is completed, vector processing continues at the next vector step. However, the switch continues to collect any digits that the caller subsequently dials until the TTR disconnects. See Collecting Digits on the switch on page 419 for more information. These “dialed-ahead” digits are saved for processing by subsequent **`collect digits`** commands. Dial-Ahead Digits are explained fully in Dial-ahead digits — collect digits command.

Functions and examples

Call Prompting uses some of the functions found in Basic Call Vectoring. Call Prompting also provides some additional functions that involve digit processing. These functions include the following:

- Treating digits as a destination
- Using digits to collect branching information (including Vector Routing Tables)
- Using digits to select options
- Displaying digits on the agent’s set
- Passing digits to an adjunct
- Creating Service Observing vectors

These functions are illustrated in the following sections.

Treating digits as a destination

Call Prompting allows you to route calls according to the digits that are collected from the caller. Once the digits are collected by the **`collect digits`** command, the **`route-to digits`** command attempts to route the call to the destination that the digits represent. The command always routes the call to the destination that is indicated by the digits that are processed by the most recent collect digits command.

The digits can represent any of the following destinations:

- Internal (local) extension, for example, split/hunt group, station, and announcement
- VDN extension
- Attendant
- Remote access extension
- External number, such as a trunk access code (TAC) or an Automatic Alternate Route/Automatic Route Selection (AAR/ARS) feature access code (FAC) followed by a public network number, for example, 7 digit ETN, 10 digit DDD.

The following example shows how a call is routed by digits that are collected from a caller.

Using Call Prompting to route by collected digits

```
1.   wait-time 0 seconds hearing ringback
2.   collect 5 digits after announcement 300
     ("You have reached Redux Electric in Glenrock.
     Please dial a 5-digit extension or wait for the
     attendant.'')
3.   route-to digits with coverage y
4.   route-to number 0 with cov n if unconditionally
5.   stop
```

In this vector, the caller is prompted to enter the destination extension of the party that he or she would like to reach (step 2). The extension in this vector may contain up to 5 digits. The vector collects the digits and then routes to the destination by the **`route-to digits`** command in step 3.

If the **`route-to digits`** command fails because the caller fails to enter any digits, or because the extension number entered is invalid, the **`route-to number`** command in step 4 routes the call to the attendant, which is the default routing option. However, as long as the destination is a valid extension, the **`route-to digits`** command succeeds, coverage applies, and vector processing terminates. If the destination is busy, vector processing terminates because coverage call processing takes effect.

Note:

Occasionally, all of the system's TTRs might be in use. As a result, when you are collecting digits from a caller, you should avoid starting your main vector with a **`collect digits`** command, since the caller in this case receives no audible feedback if he or she has to wait for a TTR to become available. Accordingly, it is a good practice to include some treatment, for example, **`wait-time 0 seconds hearing ringback`**, before the initial **`collect digits`** step.

In addition, if calls are likely to be transferred to this vector, a wait-time step of sufficient length is recommended before the collect step to allow the transferring party enough time to complete the transfer.

Using digits to collect branching information

Call Prompting allows you to direct a call to another step or vector based on the digits that are entered by the caller. This branching is accomplished with a **`goto`** step. For example, in the following vector example, digits are used to route calls to different vectors based on an assigned customer number.

Using Call Prompting to branch by collected digits

```
1.   wait-time 0 seconds hearing ringback
2.   collect 5 digits after announcement 200
     (“Please enter your customer number)
3.   goto vector 8 if digits = 10+
4.   goto vector 9 if digits = 11+
5.   goto vector 10 if digits = 12+
6.   route-to number 0 with cov n if unconditionally
7.   stop
```

The wildcard “+” indicates that the two digits can be followed by zero or any number of additional digits. Callers with a number that begins with the digits 10 are routed to vector 8, callers with a number that begins with the digits 11 are routed to vector 9, and callers with a number that begins with the digits 12 are routed to vector 10.

Vector Routing Tables

You also can test digits against entries in a Vector Routing Table.

Vector Routing Tables contain lists of numbers that can be used to test a **`goto...if digits`** command. Digits that are collected with the collect digits step can be tested to see if they are either in or not-in the specified table. Entries in the tables can include either the “+” or “?” wildcard.

- The “+” represents a group of digits and can only be used as the first or last character of the string.
- The “?” represents a single digit. Any number of them can be used at any position in the digit string.

Tables are entered on the Vector Routing Table form. See *Avaya MultiVantage Call Center Software – Guide to ACD Call Center*, 555-230-716, for complete instructions for creating Vector Routing Tables.

The following example shows a Vector Routing Table.

Vector Routing Table

```
                   VECTOR ROUTING TABLE
Number: 10         Name: Premier Accts                         Sort? n


 1:                5734020          17:                  2679038
 2:                8910573          18:                  1345+
 3:                8738494          19:                  2345+
 4:                4385702          20: _______
 5:                8768995          21: _______
 6:                7867387          22: _______
 7:                7802452          23: _______
 8:                7074589          24: _______
 9:                5674902          25: _______
10:                8789689          26: _______
11:                4870985          27: _______
12:                8093182          28: _______
13:                7809130          29: _______
14:                7890301          30: _______
15:                7893213          31: _______
16:                8743180          32: _______
```

The following Call Vector example could be used to test against the numbers provided in the Vector Routing Table.

Testing for digits in Vector Routing Table

```
1.   wait-time 0 seconds hearing ringback
2.   collect 7 digits after announcement 200 (“Please
     enter your account number”)
3.   goto vector 8 if digits in table 10
4.   queue-to split 5 pri l
5.   wait-time 10 seconds hearing ringback
6.   announcement 2771
7.   wait-time 10 seconds hearing music
8.   goto step 6 if unconditionally
```

If the caller enters an account number that is listed in the Vector Routing Table, the call is routed to vector 8. If the caller enters an account number that matches the wildcard entry (for example 1345987), the call is routed to vector 8.

If the caller enters an account number that is not listed in the Vector Routing Table, or if the caller does not enter an account number, the call is queued to split 5.

Suppose that, instead of containing a list of premier accounts, the Vector Routing Table contains a list of accounts with a poor payment record. The following example shows a vector that only queues calls with account numbers that are not in the table. Calls in the table route to the collection department.

Testing for digits not in Vector Routing Table

```
 1.  wait-time 0 seconds hearing ringback
 2.  collect 7 digits after announcement 200
     ("Please enter your account number)
 3.  goto step 11 if digits = none
 4.  goto step 6 if digits not-in table 10
 5.  route-to number 83456 with cov y if unconditionally
     (collections)
 6.  queue-to split 5 pri l
 7.  wait-time 10 seconds hearing ringback
 8.  announcement 2771
 9.  wait-time 10 seconds hearing music
10.  goto step 8 if unconditionally
11.  route-to number 0 with cov n if unconditionally
12.  stop
```

If no digits are collected, the call is routed to the operator.

Note:

Entries in Vector Routing Tables also can be tested against the telephone number of the caller Automatic Number Identification (ANI). See ANI /II-digits routing and Caller Information Forwarding (CINFO) on page 139 for more information.

Using digits to select options

Call Prompting makes it possible to provide a menu of options that the caller can use to satisfy his or her information needs. The caller selects the desired option by entering the appropriate requested digit. Once the digit is entered, a conditional branch to the appropriate treatment is made. The treatment is usually provided by the **`route-to number`** command.

The following example shows how digits are used to select options.

Using Call Prompting to select options

```
1.   wait-time 0 seconds hearing ringback
2.   collect 1 digits after announcement 3531 (Thank you
     for calling Bug Out Exterminators.  If you
     wish to learn about the services we provide, please
     dial 1.  If you'd like to set up an appointment for
     one of our representatives to visit your home or
     place of business, please dial 2.)
3.   route-to number 4101 with cov y if digit = 1
4.   route-to number 4102 with cov y if digit = 2
5.   route-to number 0 with cov n if unconditionally
6.   disconnect after announcement none
```

In step 2 of this vector, the user is asked to enter either 1 or 2, depending on the service he or she uses. If one of these digits is entered, the appropriate one of the next two steps (3 through 4) routes the call to the relevant extension, that is, either 4101 or 4102. If one of the digits is not entered, the call is routed to the attendant (step 5).

Displaying digits on the agent's set

A CALLR-INFO button can be included at the agents' display stations to help process calls that are serviced by the Call Prompting feature. However, if the agent has a two-line display set, and the display is in normal or inspect mode, the collected digits are automatically displayed on the second line. These digits remain on this line until they are overwritten, even after the call is released by the agent. On the other hand, for other display sets, the agent must press the CALLR-INFO button to display the collected digits.

It may be beneficial to install the CALLR-INFO button if you want to expedite calls by reducing the amount of time agents spend on the telephone. For example, the button could be set up to collect specific information such as a customer account number before the call is answered by the agent, thus eliminating the need for the agent to ask for this information.

The CALLR-INFO button displays information in the following format:

```
x = Info: 1234567890
```

where:

- *`x`* is a call appearance letter, for example, a, b, c, and so forth
- *`1234567890`* represents the digits that are collected from the caller

The digits that are entered by the caller are collected by the most recent collect digits command. Any digits that were dialed ahead and not explicitly requested by the most recently executed **`collect digits`** command are not displayed.

Assume that digits have been collected by Call Prompting. If the agent presses the CALLR-INFO button when the call rings at the agent station or when the station is active on a call appearance, the following events occur:

- The 10-second timer for display interval is set.
- The status lamp (if available) that is associated with the button is lit.
- The display is updated. Specifically, the incoming call identification (calling party ICI) is replaced with the collected digits in the format that was presented earlier in this section. Only those digits that were collected for the last **`collect digits`** command are displayed.

If all the conditions to use the button (except for the collection of digits) are set, and the agent presses the button, the status lamp (if available) that is associated with the button flashes denial.

One or more events may occur during a successful execution after the button is pushed. These events include the following:

- The 10-second timer times out.
- The incoming call arrives at any call appearance.
- An active call changes status, for example, another caller is added to the conference.

If any of these events occur, the following takes place:

- The status lamp (if available) that is associated with the button is turned off.
- The display is updated as previously described.

Note:

If the agent needs to display the collected digits again, the CALLR-INFO button can be pressed again to repeat the operation that is described in this section, provided that the agent is active on the call or the call is still ringing. Also, the agent can flip between the collected digits and the ICI by alternately pressing the CALLR-INFO and NORMAL buttons.

Passing digits to an adjunct

Call Prompting allows for the passing of information in the form of collected digits to an adjunct for further processing. Digits are passed to the adjunct by the ASAI Adjunct Routing capability.

An adjunct is any processor that is connected to a switch by the ASAI link. The adjunct makes a routing decision using the **`adjunct routing`** command according to caller information and/or agent availability, and it returns the routing response to the switch. For example, the adjunct can indicate that the call be routed to a specific ACD agent. This is known as direct agent calling.

A maximum of 16 Call Prompting digits from the last **`collect digits`** command can be passed to the adjunct by the **`adjunct routing`** command.

The following example, shows how Call Prompting digits are passed to an adjunct.

Using Call Prompting to pass digits to an adjunct

```
1.   wait-time 0 seconds hearing ringback
2.   collect 10 digits after announcement 300
     (‘‘Please enter your 10-digit account number.’’)
3.   adjunct routing link 50000
4.   wait-time 10 seconds hearing music
5.   route-to number 52000 with cov y if unconditionally
6.   stop
```

In step 2 of this vector, the caller is asked to enter a 10-digit account number. Once the account number is entered, the adjunct receives this information from the **`adjunct routing`** command in step 3. This command then makes the appropriate routing decision if it is able to do so. If the command succeeds within the specified wait time, the command routes the call to the appropriate destination, and the call leaves vector processing. If the command fails, vector processing continues at the next step.

In addition to the Adjunct Routing capability, collected digits also can be passed by way of ASAI to an adjunct by prompting for the digits in one vector and then routing the call to a VDN that is monitored by an Event Notification (VDN) association. The collected digits (up to 16) are sent to the adjunct in a Call Offered to Domain Event Report. See *Avaya MultiVantage CallVisor ASAI Technical Reference*, 555-230-220, for detailed information.

Note:

Adjunct Routing is fully discussed in Adjunct (ASAI) Routing on page 163.

Creating Service Observing vectors

Service Observing vectors can be constructed to allow users to observe calls from a remote location or local station. When combined with Call Prompting, Service Observing vectors can route calls to:

- A Remote Access extension
- A Service Observing Feature Access Code (FAC) and extension that is entered by the user
- A preprogrammed FAC and extension

Remote access Service Observing vector

The following vector example connects a user to Remote Access. Once connected, the user can dial either a listen-only or listen/talk Service Observing FAC followed by the extension number to be observed. Although it is not required, Call Prompting increases security by providing passcode protection with remote service observing.

Remote access Service Observing vector

```
1.   wait-time 0 secs hearing ringback
2.   collect 5 digits after announcement 2300
     (“Please enter your 5-digit security code.”)
3.   goto step 5 if digits = 12345 (security code)
4.   disconnect after announcement 2000
5.   route-to number 5000 with cov n if unconditionally
6.   stop
```

User-entered FAC and extension

The following vector example connects a user directly to the Service Observing FAC and extension based on the digits that are collected by Call Prompting.

Service Observing vector with user-entered FAC and extension

```
1.   wait-time 0 secs hearing ringback
2.   collect 5 digits after announcement 2300
     (‘‘Please enter your 5-digit security code.’’)
3.   goto step 5 if digits = 12345 (security code)
4.   disconnect after announcement 2000
5.   wait-time 0 seconds hearing ringback
6.   collect 6 digits after announcement 3245 (“Please
     enter the number 11 for listen-only
     observing or the number 12 for listen/talk observing
     followed by the number of the extension you would
     like to observe”)
7.   route-to digits with coverage n
8.   stop
```

Pre-programmed FAC and extension

The following example shows a vector that connects a user to a pre-programmed FAC and extension using Call Prompting to allow the observer to select the extension that he or she wants to observe. In this example, the observer will be Service Observing a VDN.

Service Observing vector with programmed FAC and extension

```
1.   wait-time 0 secs hearing ringback
2.   collect 5 digits after announcement 2300
     (‘‘Please enter your 5-digit security code.’’)
3.   goto step 5 if digits = 12345 (security code)
4.   disconnect after announcement 2000
5.   wait-time 0 seconds hearing ringback
6.   collect 1 digits after announcement 2310 (“Enter 1 to
     observe sales, 2 to observe billing”)
7.   route-to number 113001 with cov n if digit = 1
     (11 = listen-only observe, 3001 = “Sales” VDN)
8.   route-to number 113002 with cov n if digit = 2
     (11 = listen-only observe, 3002 = “Billing” VDN)
9.   goto step 6 if unconditionally
```

Dial-ahead digits — collect digits command

Dial-ahead digits provide the caller with a means of bypassing unwanted announcement prompts on the way to acquiring the information or servicing he or she wants. These digits are available for use only by subsequent **`collect digits`** commands. The digits are never used by other vector commands that operate on digits, for example, **`route-to digits`**, and **`goto...if digits`**, until they are collected. These digits are not forwarded with interflowed calls. In addition, these digits are not displayed as part of the CALLR-INFO button operation until they are collected by a **`collect digits`** command.

Collection of dial-ahead digits continues until one of the following occurs:

- Vector processing stops or is terminated.
- The sum of the digits collected for the current **`collect digits`** command plus the dial-ahead digits exceeds the switch storage limit of 24. Any additional digits are discarded until additional storage is made available by a subsequent **`collect digits`** command.

  Note:

  Any asterisk (*) and pound sign (#) digits that are dialed ahead count toward the 24 digit limit, as do any dial-ahead digits that are entered after the asterisk or pound sign digit.

- The TTR required by the user to collect digits is disconnected. This happens whenever one of the following conditions is true:
  - — A successful or unsuccessful **`route-to number`** step is encountered during vector processing, except where the number routed to is a VDN extension.
  - — A successful or unsuccessful **`route-to digits`** step is encountered during vector processing, except where the number routed to is a VDN extension.
  - — A successful or unsuccessful **`adjunct routing`** step is encountered during vector processing.
  - — A successful or unsuccessful **`converse-on`** step is encountered during vector processing.
  - — A Call Prompting timeout occurs, during which time the caller has not dialed any additional digits, asterisks (*) or pound signs (#).
  - — Vector processing stops or is terminated.
  - — A successful or unsuccessful collect ced/cdpd step is encountered.

  Note:

  When the TTR is disconnected due to a **`route-to number`**, or **`route-to digits`**, **`converse-on`**, **`adjunct routing`**, or **`collect ced/cdpd`** step, all dial-ahead digits are discarded. This means that following a failed **`route-to`**, **`converse`**, or **`adjunct routing`** step, a subsequent **`collect digits`** step always requires the user to enter digits.

Dial-ahead digit vector examples

The vectors shown in the following examples illustrate a situation where a caller can enter dial-ahead digits. In this case, the caller is required to have a touch-tone telephone. An alternative handling sequence should be programmed in case the caller has a rotary telephone or the caller does not dial a touch tone digit before the timeout period.

Step 2 of Vector 30 gives the caller two options, each of which provides different information. The caller is prompted to enter either 1 or 2, depending on what information he or she wants to hear. Once the caller enters a digit, the digit is collected by the **`collect digits`** command. Thereafter, an attempt is made by the **`route-to number`** command to route the call to the appropriate vector (step 3 or 4). If the caller enters a digit other than 1 or 2, the appropriate announcement is provided (step 5), and the digit entry cycle is repeated (step 6).

If the caller enters 1, Vector 31 is accessed.

Using dial-ahead digits to bypass announcements, example 1

```
VDN (extension=1030   name=''Coastal''   vector=30)
Vector 30:
1.   wait-time 0 seconds hearing ringback
2. collect 1 digits after announcement 3000
     (''Thank you for calling Coastal League Baseball
     Hotline. You must have a touch-tone telephone to use
     this service. If you wish to hear the scores of
     yesterday's games, please press 1.  If you wish to
     hear today's schedule of games, please press 2.'')
3. route-to number 1031 with cov y if digit = 1
4. route to number 1032 with cov y if digit = 2
5. announcement 301 (''Entry not understood.  Please
     try again.'')
6. goto step 2 if unconditionally
```

In step 1 of Vector 31 (below), the caller is given three options that supplement the original option that was provided in Vector 30. The caller is prompted to enter either **3**, **4**, or **5**, depending on what information he or she wants to hear. If the caller enters an incorrect digit, the customary digit correction routine is implemented (steps 5 and 6). Once an appropriate digit is entered, the call is routed, in this example by a **`goto step`** command (step 2, 3, or 4), to the appropriate announcement (step 7 or step 9).

In step 10 of Vector 31, the caller is prompted with the choice of returning to the main menu provided in Vector 30 or of terminating the call. If the caller selects the former option (by entering 9), the call is routed to Vector 30, and the entire process is repeated.

Using dial-ahead digits to bypass announcements, example 2

```
VDN (extension=1031 name=‘‘Scores’’   vector=31)
Vector 31:
1.   collect 1 digits after announcement 4000
     (‘‘If you wish to hear scores of games in both divisions,
     please press 3.  If you wish to hear scores for Northern
     Division games only, please press 4.  If you wish to hear
     scores for Southern Division games only, please press 5.’’)
2. goto step 7 if digits = 3
3. goto step 7 if digits = 4
4. goto step 9 if digits = 5
5. announcement 301 (‘‘Entry not understood.  Please
     try again.’’)
6. goto step 1 if unconditionally
7. announcement 4002 (Northern Division scores)
8. goto step 10 if digits = 4
9. announcement 4003 (Southern Division scores)
10. collect 1 digits after announcement 4004
     (‘‘If you wish to return to the main menu, please press 9.
     Otherwise, press 0.)
11. route-to number 1030 with cov n if digit = 9
12. goto step 15 if digit = 0
13. announcement 301 (’’Entry not understood. Please try again.‘‘)
14. goto step 10 if unconditionally
15. disconnect after announcement none
```

Vector 32 (below) is similar in design to Vector 31. The major difference is the information provided and the requested digit entries.

In this example, the caller has to go through at least two sets of options to get the information that he or she wants. Each option set is introduced by an announcement. However, because of the dial-ahead digit capability, the caller can bypass the announcements if he or she chooses. Thus, the caller could enter 1 and 5 within a matter of seconds to hear yesterday’s Southern Division scores.

The caller may enter digits while he or she is being queued for an announcement or while the announcement is playing. If digits are entered during an announcement, the announcement is disconnected. If digits are entered while a call is queued for an announcement, the call is removed from the announcement queue.

Dial-ahead digits, example 2

```
VDN (extension=1032   name=Schedule   vector=32)
Vector 32
1.   collect 1 digits after announcement 5000
     (‘‘If you wish to hear today’s schedule of games in
     both divisions, please press 6.  If you wish to
     hear today’s schedule of games in the Northern
     Division only, please press 7.  If you wish to hear
     today’s schedule of games in the Southern Division
     only, please press 8.’’)
2.   goto step 7 if digits = 6
3.   goto step 7 if digits = 7
4.   goto step 9 if digits = 8
5.   announcement 301 (‘‘Entry not understood.  Please
     try again.’’)
6.   goto step 1 if unconditionally
7.   announcement 5002 (Northern Division schedule)
8.   goto step 10 if digits = 7
9.   announcement 5003 (Southern Division schedule)
10.  collect 1 digits after announcement 4004
     (‘‘If you wish to return to the main menu,
     please press 9.  Otherwise, press 0.)
11.  route-to number 1030 with cov n if digit = 9
12.  goto step 15 if digits = 0
13.  announcement 301 (’’Entry not understood.  Please
     try again.‘‘)
14.goto step 10 if unconditionally
15.disconnect after announcement none
```

ASAI-requested digit collection

The ASAI-requested digit collection feature gives an adjunct the ability to request that a DTMF tone detector be connected for the purpose of detecting user-entered digits. The digits that are collected as a result of this feature are passed to ASAI monitoring and/or controlling adjuncts for action. The switch handles these digits as if they were dial-ahead digits. This feature allows the caller to request Sequence Dialing after the call has been routed to the final destination and has resulted in an unanswered call, that is busy, no answer, and so forth.

These digits are not necessarily collected while the call is in vector processing. They are sent to an ASAI adjunct, or they may be used by Call Prompting features, or both.

ASAI Adjunct Routing and Call Prompting features must be enabled on the switch for this feature to work.

ASAI-provided dial-ahead digits — collect digits command

The ASAI-provided digits feature allows an adjunct to include digits in a Route Select capability. These digits are treated as dial-ahead digits for the call. Dial-ahead digits are stored in a dial-ahead digit buffer and can be collected (one at a time or in groups) using the **`collect digits`** command(s). Although the adjunct may send more than 24 digits in a Route Select, only the first 24 digits (or 24-*x*, where *x* is the number of digits that are collected by vector processing prior to executing the **`adjunct routing`** vector command) are retained as dial-ahead digits. An application can use this capability to specify the digits that the switch should pass to the VRU as part of the **`converse-on`** vector step.

Note:

The maximum number of dial-ahead digits that can be stored in the buffer is dependent on the number of digits that were already collected for the call by a previous **`collect digits`** vector command. If *x* digits were collected by vector processing prior to executing an **`adjunct routing`** vector command, the *x* digits collected reduces the maximum number of digits that can be stored as dial-ahead digits as a result of a Route Select. The rest are discarded.

Considerations

You should keep the following considerations in mind when working with Call Prompting:

- To enter the digits requested via a **collect digits** command, outside callers must have a touch-tone telephone. For such callers using rotary dialing, a 10 second inter-digit timeout takes effect, and the **collect digits** command is omitted. As a precaution, a default treatment (for example, **route-to attendant** command, **queue-to split** command) should always be provided in the vector script unless the script is created exclusively for users of touch-tone telephones.
- If a caller does not enter the full number of digits specified in a **collect digits** step, an administered timeout occurs. Thereafter, vector processing continues with subsequent vector steps, and an attempt is made to process the call using the digits that have been collected. If the digits entered do not represent a valid destination, and if Automated Attendant is being implemented via a **route-to digits** command, the **route-to digits** command fails, and vector processing continues at the next step, which should be a default treatment.
- It may be prudent to take steps in case a **route-to attendant** command fails, such as providing a disconnect announcement.
- From time to time, all of the system's touch-tone receivers might be in use. As a result, you should avoid starting your main vector with a **collect digits** command, since the caller on a DID or tie trunk in this case receives no audible feedback if he or she has to wait for a receiver to become available. Accordingly, it is a good practice to include some treatment (for example, a **wait-time 0 seconds hearing ringback** step) before the initial **collect digits** step. The **wait-time** step is not necessary if the collect step is collecting ced or cdpd digits.

Chapter 11: Look-Ahead Interflow (LAI)

Look-Ahead Interflow (LAI) enhances Call Vectoring for contact centers with multiple ACD locations. LAI allows these centers to improve call-handling capability and agent productivity by intelligently routing calls among contact centers to achieve an improved ACD load balance. This service is provided by ISDN D-channel messaging over QSIG or non-QSIG private networks, virtual private networks, or public networks. The receiving switch is able to accept or deny interflowed calls sent by the sending switch.

LAI has the following basic attributes:

- Produces First in First Out (FIFO) or near-FIFO call processing
- Includes enhanced information forwarding, that is, codeset 0 user information transport

This chapter includes the following topics:

LAI prerequisites

The following items are criteria for basic LAI call control operation over a virtual private network or a public switched network:

- The sending and receiving contact center locations must have ISDN (PRI or BRI) trunk facilities.

  Note:

  ATM trunking and IP trunking can be set up to emulate ISDN PRI. For information on setting this up, see *Administration for Network Connectivity for Avaya MultiVantage Software*, 555-233-504, and *ATM Installation, Upgrades and Administration for Avaya DEFINITY Servers*, 555-233-124.

- The switch must support the ISDN country protocol.
- LAI has been tested with several major carriers. To find out if these capabilities work with your carrier, check with your account team for the most current information. If testing has not been done to verify operation over the public networks that are involved with the preferred specific configuration, use of private ISDN trunking between the nodes should be assumed until successful testing is complete.
- The ISDN SETUP and DISCONNECT messages are transported between sending and receiving locations, for example, SS7 or equivalent public network connectivity.
- A receiving-end generated DISCONNECT message must transmit back to the sending the switch contact center without changing the cause value.

  Conversion of the DISCONNECT message to a progress message (with a Progress Indicator Description set to 1 and a Cause Value other than 127 included) is a valid reject message and compatible with LAI.

- Progress messages that are generated towards the sending end by intervening network switches must have the Progress Indicator Description set to 8 so that the switch does not consider the call accepted or rejected.
- ISDN codeset 0 user information transport supports LAI information forwarding. As an alternative, LAI can use dedicated VDNs at the receiving location to provide an equivalent display of the forwarding application identity and set trunk group options to not send either the codeset 6/7 LAI IE or codeset 0 information transport.

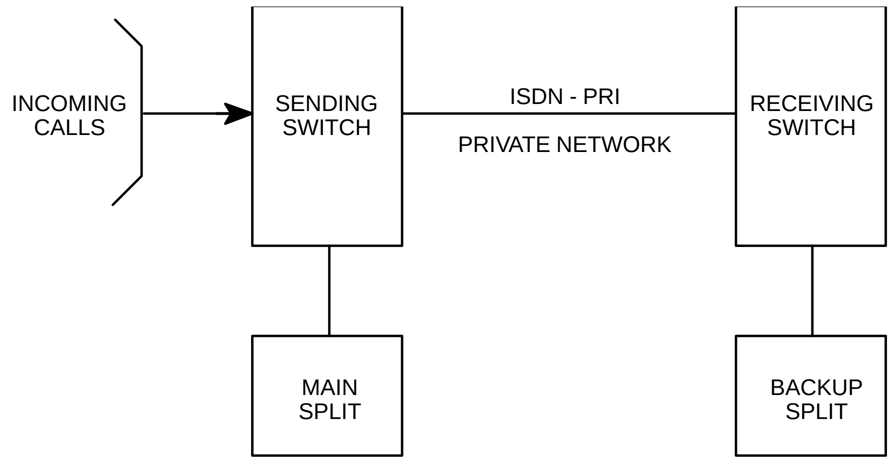
  Note:

  Best Service Routing (BSR) cannot use these LAI alternatives. BSR must use ISDN codeset 0 user information transport.

Example of a two-switch configuration

Look-Ahead Interflow (LAI) is enabled through the use of call vectors and their associated commands. For a two-switch configuration, these vectors are included in both the sending switch, which processes vector outflow, and the receiving switch, which processes vector inflow. The two-switch configuration for LAI is shown in the following figure.

Two-switch LAI connections



Command set

LAI enhances call vectoring so that calls interflow only to those remote locations that can accept the calls.

LAI is achieved through a set of vector commands. The following table lists the call-acceptance vector commands that are used in LAI.

Call-acceptance vector commands

Command	Qualification
announcement	Announcement available Queued for announcement Retrying announcement
check split	Call terminates to agent Call queued to split
collect digits	Always (except for ced and cdpd digits, which are neutral)
converse-on split	VRU answers the call Call queued to converse split
disconnect	With announcement and announcement available With announcement and queued for announcement With announcement and retrying announcement
messaging split	Command successful Call queued
queue-to split	Call terminates to agent Call queued to split
route-to	Terminates to valid local destination Successfully seizes a non-PRI trunk Results in a LAI call attempt, and the call is accepted by the far-end switch
wait-time	Always (except **wait-time hearing i-silent**, which is neutral)

If the receiving switch decides it is unable to accept the LAI call, call denial is accomplished by executing one of the vector commands that are listed in the following table.

Note:

It is recommended that you use **busy** instead of **disconnect** to allow for compatibility with similar network services such as Alternate Destination Redirection (ADR).

Call-denial vector commands

Command	Qualification
busy	Always
disconnect	Without announcement With announcement but announcement unavailable
reply-best	Always; used with BSR

The vector commands that are shown in the following table are considered neutral because they do not generate either call acceptance or denial messages.

Neutral vector commands

Command	Qualification
adjunct routing	Always
announcement	Announcement unavailable
check split	Call neither terminates nor queues
collect ced/cdpd digits	Always
consider	Always - used with BSR
converse-on split	Call neither terminates nor queues
goto step	Always
goto vector	Always
messaging split	Command failure
queue-to split	Call neither terminates nor queues

Neutral vector commands (continued)

Command	Qualification
`route-to`	Unsuccessful termination Trunk not seized LAI call denied by the far-end switch
`stop`	• Always
`wait-time hearing i-silent`	• Always **Note:** This command is used following an **`adjunct routing`** command in applications where the adjunct decides whether to accept or reject the Look-Ahead calls.

How traditional LAI works

Traditional LAI is recommended when the preferred call flow performs LAI attempts before queuing the call.

LAI uses the commands that are included within the Basic Call Vectoring and Call Prompting features:

- **`route-to number with coverage n`** or **`route-to digits with coverage n`** command on a switch that has LAI enabled and that successfully seizes an ISDN trunk automatically results in a normal LAI call attempt being placed. The call attempt can be rejected or accepted by the remote end.
- **`route-to number with coverage y`** or **`route-to digits with coverage y`** command never results in a LAI call attempt. The sending end assumes that the call is always going to be accepted. This command always completes the call. Moreover, the command should not be used when the vector at the receiving location ends up denying the call, since the caller in this case is given a busy signal, or the call is disconnected. Use this command with coverage set to **y** only for those cases when an unconditional interflow is wanted (with LAI active) and the terminating switch is set up to handle this type of call.

When a LAI call attempt is made, Call Vectoring at the sending location checks a potential receiving location to determine whether to hold or send the call. While this is done, the call remains in queue at the sending location. As such, the call can still be connected to the sending-location agent if one becomes available before the receiving location accepts the call.

Call Vectoring at the receiving location decides whether to accept the call from the sending location or to instruct the sending location to keep the call. In the latter case, the sending location can then either keep the call, check other locations, or provide some other treatment for the call. Conditions for sending, refusing, or receiving a LAI call attempt can include a combination of any of the following:

- Expected wait time for a split
- Number of staffed or available agents
- Number of calls in queue
- Average speed of answer or the number of calls active in a VDN
- Time of day and day of week
- Any other legitimate conditional

If the call is accepted by the receiving switch, the call is removed from any queues at the sending switch, and call control is passed to the receiving switch. If the call is denied by the receiving switch, vector processing continues at the next step at the sending switch. Until the call is accepted by either switch, the caller continues to hear any tones applied by the sending switch. If the call is denied, the call vector can apply alternate treatment, such as placing another LAI call to an alternate backup switch.

Note:

The LAI operation is completely transparent to the caller. While a LAI call attempt is being made, the caller continues to hear any audible feedback that is provided by the sending switch vector. The caller also maintains his or her position in any split queues until the call is accepted at the receiving switch.

LAI passes Call Prompting digits collected in the sending switch to the receiving switch by codeset 0 user information transport. For more information, see Information Forwarding on page 151.

Example of traditional LAI

The vectors in the sending switch use the **goto** command to determine whether the call should be sent to the receiving switch. Recall that the **goto** command tests various outflow threshold conditions such as expected wait time. If the expressed condition is met, a branch is made to the appropriate **route-to** command. This command sends the call to the receiving switch, which, as already noted, can accept or deny the call.

The following example shows an outflow vector that might be included in a sending switch.

Using LAI with route-to commands to outflow calls

```
1.   wait-time 0 secs hearing ringback
2.   goto step 5 if expected-wait for split 3 pri m < 30
3.   route-to number 5000 with cov n if unconditionally
4.   route-to number 95016781234 with cov n if unconditionally
5.   queue-to split 3 pri m
6.   announcement 3001
7.   wait-time 30 secs hearing music
8.   goto step 6 if unconditionally
```

If split 3 has an expected wait time of less than 30 seconds (step 2), step 5 queues the call to the split’s queue at a medium priority.

If the expected wait time is 30 seconds or more, LAI attempts are made in steps 3 and 4. If the call is accepted by one of the receiving switches call control passes to the receiving switch.

If the receiving switches deny the call, the call queues to split 3 and announcement 3001 plays. The caller then hears music (interrupted by announcement 3001 every 30 seconds).

Receiving switch operation

When the receiving switch receives the LAI request, the call first routes to a VDN. The VDN then maps the call to the receiving switch's inflow vector, and vector processing begins, starting with inflow checking. Inflow checking is enabled by conditional **`goto`** commands in the inflow vector. The decision to accept or deny a call can be based on checks such as any of the following:

- Expected Wait Time
- Number of staffed agents
- Number of available agents
- Time-of-day/day of the week
- Number of calls in split's queue
- Average Speed of Answer
- Active VDN Calls
- ANI
- II-Digits
- CINFO ced and/or cdpd digits
- Collected digits forwarded from the sending switch

Once inflow checking is complete, acceptance of the LAI call is accomplished by executing any of the vector commands listed in Call-acceptance vector commands on page 206.

Note:

For each of the commands listed in Call-acceptance vector commands on page 206, Neutral vector commands on page 207 and Call-denial vector commands on page 207, only one of the corresponding qualifications needs to be true for the command to effect the desired result, which is call acceptance, call denial, or no effect on such acceptance or denial.

The following example shows an inflow vector that might be used by a receiving switch.

Using inflow checking for LAI requests

```
1.   goto step 6 if expected-wait in split 1 pri h > 30
2.   queue-to split 1 pri h
3.   announcement 4000
4.   wait-time 2 seconds hearing music
5.   stop
6.   busy
```

Step 1 of this inflow vector checks the inflow thresholds. The **`goto step`** command in step 1 checks the expected wait time in split 1. If the expected wait time is greater than 30 seconds, a branch is made to the **`busy`** command in step 6. If executed, the **`busy`** command denies the call, and the receiving switch returns a call denial message to the sending switch. The sending switch, in turn, drops the LAI call attempt and then continues vector processing at the next vector step.

If the expected wait time in split 1 is less than or equal to 30 seconds, the receiving switch returns a call acceptance message to the sending switch, and call control is passed to the receiving switch. Thereafter, the call is queued to split 1 in the receiving switch (step 2). Once queued, the caller receives the appropriate announcement in step 3 and is then provided with music until the call is answered by an agent or abandoned by the caller (steps 4 and 5). Remember that the **`stop`** command halts vector processing but does not drop the call.

If the sending switch does not receive a call acceptance or call denial message within 120 seconds after the LAI call request, the LAI attempt is dropped. The sending switch continues vector processing at the next step.

How enhanced LAI works

Enhanced LAI uses the same basic vectoring commands as traditional LAI, but adds the conditional **`interflow-qpos`**. Enhanced LAI is recommended when the preferred call flow performs LAI attempts after queuing the call.

Using enhanced LAI **`interflow-qpos`** conditional:

- Produces First in First Out (FIFO) or near FIFO call processing
- Uses less processing during LAI

The simple way to achieve FIFO

You can use the **`interflow-qpos`** conditional in a **`route-to`** or **`goto`** command to achieve FIFO results.

For example, you can use the following **`route-to`** command with the conditional to achieve FIFO results:

```
route-to number 9581234 with cov n if interflow-qpos=1
```

If you have a lot of remote agents, you may want to set the **`route-to`** command as follows:

```
route-to number 9581234 with cov n if interflow-qpos<=2
```

Detailed information about the interflow-qpos conditional

You can use this feature without understanding the differences between split queues and eligible queues or between **`interflow-qpos`** and queue position. There are features that are built into enhanced LAI so that when you write a step such as **`route-to number 9581234 with cov n if interflow-qpos=1`**, the system operates smoothly under all conditions.

The interflow-qpos conditional

The **`interflow-qpos`** conditional only applies interflow processes to a dynamic eligible queue and to calls that are queued locally before the **`route-to`** is attempted.

The eligible queue is that portion of the split/skill queue that:

- Includes only calls that are not expected to be answered locally during the interflow process at that moment relative to the call being processed
- Does not include direct agent calls because these calls are excluded from any interflow process.

The following is an example of the **`interflow-qpos`** conditional used in a **`route-to`** command:

`route-to number _____ with cov _ if interflow-qpos CM x`

where

- ***`CM`*** is the comparator. It is one of three symbols: =, <, <=
  - — With **`if interflow-qpos = x`**, the call is interflowed if it is at the ***`x`*** position from the top of the eligible queue.
  - — With **`if interflow-qpos < x`**, the call is interflowed if it is among the top ***`x-1`*** of the eligible queue.
  - — With **`if interflow-qpos <= x`**, the call is interflowed if it is among the top ***`x`*** eligible calls.
- ***`x`*** indicates the call's position in the eligible queue. Valid queue positions are 1 through 9. The top queue position is 1. The eligible queue is made up of calls from the first local split/skill that the call has been queued to due to previous steps in the vector.

Note:

Calls that are likely to be serviced locally before an LAI can be completed are not eligible for interflow since they are excluded from the eligible queue. Calls that are likely to be answered are identified based on conditions of the split/skill to which the call is queued and, under certain conditions, an administered minimum EWT threshold value.

The following is an example of the **`interflow-qpos`** conditional used in a **`goto`** command:

`goto step/vector ____ if interflow-qpos CM x`

where

- ***`CM`*** is the comparator. It is one of six symbols: =, <>, <, <=, >, >=
- ***`x`*** indicates the call's position in the eligible queue. Valid queue positions are 1 through 9. The top queue position is 1.

Calls that are likely to be serviced locally before an LAI can be completed are not eligible for interflow since they are excluded from the eligible queue.

When does a call not interflow?

A call does not interflow under the following circumstances:

- If the **`interflow-qpos`** conditional is not met.

  As with other conditionals, the **`route-to number... if interflow-qpos`** step or the **`goto step/vector`** branch is executed only if the conditional is met, otherwise vector processing goes to the next step.

- If the call is not in a split/skill queue or not in the eligible portion of the queue when the conditional step is executed.

  If the call is not in queue when the **`route-to number... if interflow-qpos`** step is executed, a vector event is logged and vector processing continues at the next step.

  If the call is not in queue when a **`goto... if interflow-qpos`** step is executed, the queue position of the call is considered to be infinite in determination of the conditional.

  Note:

  A vector event is not logged if the call is in queue, but is not in the eligible portion of the queue.

- Interflow failure or LAI rejection

  Interflow failure or LAI rejection will also go to the next step. Route-to operation and feature interactions will be the same as other configurations of the route to number command, for example, **`route to number ___ with cov _ if digit`** ***`CM x`***.

The following table outlines what action is taken for different cases of interflow eligibility.

Actions taken for cases of interflow eligibility

Case	Action at route-to step	Action at goto step
The call not eligible for interflow.	The call is never routed.	Treat as if the interflow queue position is infinite.
The call is not in any split queue.	The call is treated as if the interflow queue position is infinite.	Treat as if interflow queue position is infinite.
The call is eligible for interflow.	Act according to the conditional.	Act according to the conditional.

How the minimum EWT is set

The minimum expected wait time (EWT) threshold that is used to help determine which calls are more likely to be answered locally is administered on the Feature-Related System Parameters form. Minimum EWT is used when the local agents, that is, in the first split/skill to which the call is queued, are handling a significant number of the calls. If these agents are not handling a significant number of calls, the call is eligible for LAI even if its EWT is lower than the threshold.

Note:

When enhanced LAI vectors or the look-ahead EWT threshold are administered inappropriately, remote agents may experience phantom calls or a delay between becoming available and receiving an ACD call.

The instructions below assume that you use a SAT terminal or terminal emulator to administer the switch.

To set the minimum EWT threshold:

1. In the command line, type **`change system-parameters feature`** and press **Enter**.

   The system displays the Feature-Related System Parameters form.

2. Find the page of the Feature-Related System Parameters form that has the `Interflow-Qpos EWT Threshold` field.

   If Look-Ahead Interflow is active, the `Interflow-Qpos EWT Threshold` field can be administrated.

3. In the `Interflow-Qpos EWT Threshold` field, enter the number of seconds, as a number from 0 to 9, that you want for the EWT threshold. The default of 2 seconds is recommended.

   Note:

   When the look-ahead EWT threshold field is set too low, remote agents may experience phantom calls.

4. Press **Enter** to save your changes.

Example of single-queue multi-site operation

In this scenario, all new calls for a given customer application are routed by the public network to only one of the switches in the network, where the calls are put in the queue.

Local agents service the calls from the queue in the normal fashion; however, remote agents service calls by means of enhanced look-ahead.

The switch with the call queue does rapid enhanced look-ahead attempts to all other switches in the network that can service this call type, looking for an available agent.

Normally, the look-ahead attempts are placed only on behalf of the call that is at the head of the queue (**`interflow-qpos = 1`**). However, in scenarios where there are large numbers of agents at a remote switch, it may be necessary to do interflows on behalf of more than one call in order to outflow a sufficient volume of calls to keep all agents busy (**`interflow-qpos <= 2`**).

Vector to back up split

```
1.   announcement 3501
2.   wait-time 0 secs hearing music
3.   queue-to skill 1 pri m
4.   route-to number 93031234567 with cov n if interflow-qpos = 1
5.   route-to number 99089876543 with cov n if interflow-qpos = 1
6.   wait-time 5 secs hearing music
7.   goto step 4 if unconditionally
```

In this example, interflow call attempts are placed on behalf of the call that is at the beginning of the queue every 5 seconds to the two other switches in the network.

If queuing times are very long, 5 minutes, for example, and the call is not near the beginning of the queue, it is wasteful to go through the vector loop from step 4 to step 7 every 5 seconds. For this reason, the FIFO processing vector on page 218 is more efficient.

Example of maintaining FIFO processing with LAI

One of the advantages of enhanced LAI is the ability to provide FIFO or near-FIFO call processing. The following example shows a vector that is used to achieve such call processing.

FIFO processing vector

```
1.   announcement 3501
2.   wait-time 0 secs hearing music
3.    queue-to skill 1 pri m
4.   goto step 7 if interflow-qpos < 9
5.   wait-time 30 secs hearing music
6.   goto step 5 if interflow-qpos >= 9
7.   route-to number 93031234567 with cov n if interflow-qpos = 1
8.   route-to number 99089876543 with cov n if interflow-qpos = 1
9.   wait-time 5 secs hearing music
10.  goto step 7 if unconditionally
```

In this vector:

- The rapid look-ahead loop is only entered when the call reaches one of the top 8 positions in queue.
- The number of executed vector steps is reduced dramatically when call waiting times are long.

It is important to write vectors so that calls at the head of the queue have advanced to the rapid look-ahead loop by the time their turn to interflow has been reached. In the vector example shown above, if 8 calls can be serviced from queue in less than 30 seconds (which is the loop time on step 5), there can be a delay in outflowing calls to available agents at the remote sites.

Single-queue FIFO considerations

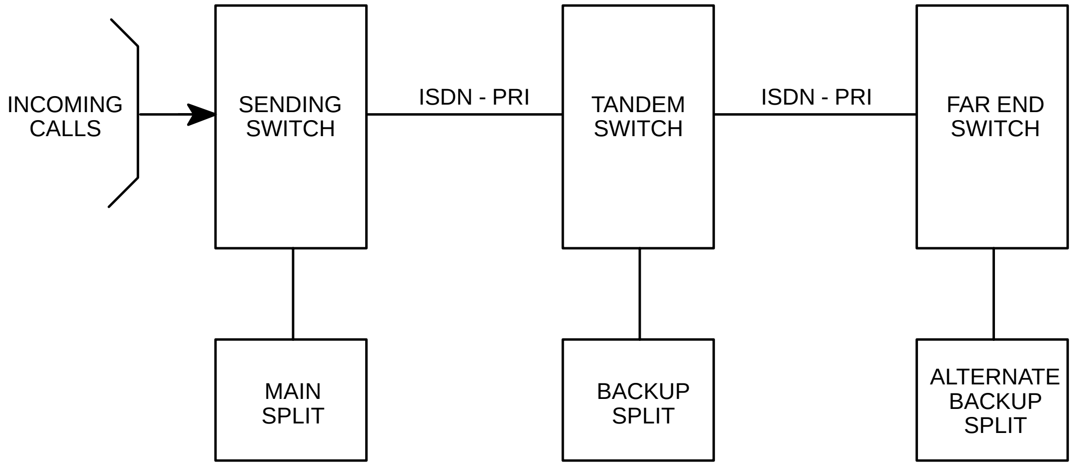
The following issues need to be taken into consideration for FIFO in a single queue:

- When there are available agents, calls are always delivered to available agents at the queuing switch before available agents at the remote switches.
- When there are calls in the queue and agents serve calls from multiple applications, the agents always service calls from the applications that are queued locally before calls from applications that are queued at another switch.
- Backup VDNs and vectors are recommended in order to provide continuous operation in the event of a failure at a queuing switch.
- EWT predictions cannot be made if the split/skill in which the calls are queued has no working agents.
- EWT predictions may be temporarily inaccurate if there are sudden, major changes in the number of working agents in the split/skill in which the calls are queued.

Example of LAI in a tandem switch configuration

Tandem LAI is implemented by using **`route-to`** commands that contain external destinations that route over ISDN facilities. This configuration is shown in the following figure.

LAI using a tandem switch



Sending switch operation

The sending switch is unaware that its LAI call is being tandemed to an alternate switch. The operation of the sending switch in the tandem switch configuration is the same as that in the two-switch configuration.

Tandem switch operation

If the receiving switch executes a **`route-to`** command that routes the call over an ISDN facility before call acceptance, the **`route-to`** command is performed on a “look-ahead” basis in the same manner as a sending switch. If the call is accepted at the far-end switch, acceptance is passed to the sending switch, and call control is passed to the far-end switch, along with tandeming of the original calling party information and the original DNIS name. If the call is denied, the next step of the tandem switch vector is executed.

The following example shows a tandem switch vector.

Tandem switch vector example

```
 1.  goto step 6 if expected-wait in split 30 pri h > 30
2.   queue-to split 30 pri h
3.   announcement 200
4.   wait-time 2 seconds hearing silence
5.   stop
6.   route-to number 4000 with cov n if unconditionally
7.   busy
```

Step 1 of this vector checks the inflow threshold. If the inflow criteria are acceptable, the vector flow drops to step 2, where the **`queue-to split`** command provides acceptance to the sending switch. Thereafter, steps 3 through 5 provide a typical queuing-wait scheme.

If the inflow criteria are not acceptable, a branch is made to step 6. The **`route-to`** command in this step checks another switch that is enabled with LAI on a look-ahead basis. If this far-end switch rejects the call, a denial message is relayed back to the tandem switch, which then drops the LAI call attempt. On the other hand, if the far-end switch accepts the call, an acceptance message is relayed all the way back to the sending switch.

No ringback is provided in this tandem switch vector. This is necessary so that an acceptance message is not returned to the sending switch. This operation is appropriate for the caller because the sending switch has already returned an announcement before a LAI attempt is made to the receiving switch.

Be sure that the sending switch is not used as a backup location for the tandem switch or for any of the far-end switches. If the sending switch is administered in this manner, all trunk facilities could be tied up by a single call.

Far-end switch operation

The far-end switch is also unaware that tandeming has taken place. The far-end switch functions in the same manner as the receiving switch within the two-switch configuration.

LAI-initiated path-replacement for calls in vector processing

Path replacement for calls in queue and vector processing can be accomplished using QSIG or DCS with Reroute using ISDN SSE. For calls that are waiting in queue or in vector processing, even if the call is not connected to an answering user, path replacement can be attempted to find a more optimal path for this call. This results in more efficient use of the trunk facilities.

The **`route-to`** command is used in LAI to initiate a QSIG path replacement for a call. The following scenario can take place:

- At the terminating DEFINITY, if a Path Replacement Propose operation is received for a call that is in queue or vector processing, the switch can immediately initiate path replacement using the Path Replacement Extension if the `Path Replace While in Queue/Vectoring` field is set to y and the `Path Replacement Extension` field has a valid entry. These fields are located on the ISDN parameters page of the Feature-Related System Parameters form.

Note:

The Call Management System (CMS) load r3v9ag and earlier cannot track a measured ACD call after path replacement has taken place. CMS load r3v9ah and newer does keep the CMS call record of the measured ACD call intact after path replacement takes place.

Example vector

The following example shows how an LAI vector can be written to trigger path-replacement at the terminating switch.

Note:

In order for a path-replacement to be attempted, the incoming and outgoing trunks that are used for the call must be administered with the `Supplementary Service Protocol` field set to b.

LAI-initiated path-replacement vector

```
1.   wait 0 seconds hearing music
2.   queue-to skill “n” if available-agents < 6
3.   route-to number “ARS number for ISDN trunk” with cov n
4.   wait 999 seconds hearing ringback
```

At the terminating (receiving) switch, the vector that is executed by the incoming call must be programmed with an **`announcement`**, or **`wait hearing music`** vector command. The use of one of these commands is what makes it possible for path-replacement to take place while the call is in vector processing.

DNIS and VDN override in an LAI environment

LAI handles Dialed Number Identification Service (DNIS) and VDN Override in various ways, depending on a number of different characteristics of the call. DNIS, as described in Call Vectoring fundamentals on page 41, allows any agent with a display-equipped telephone to receive visual displays that specify the name of the called VDN. VDN Override in its basic form allows the name of a subsequently routed to VDN to be displayed to the answering agent instead of the name of the originally called VDN.

The following sections discuss how LAI handles DNIS and VDN Override.

Answering agent's display

For LAI, the DNIS name, which is the called VDN name from the sending switch, is presented on the answering agent's display on the receiving switch if all of the following are true:

- The LAI option is enabled.
- The call routes to a VDN.
- The DNIS name field is not blank.

The type of DNIS information that is displayed depends upon a number of different scenarios. This information is presented in the following table.

DNIS information displayed for LAI scenarios

Scenario	Information displayed
Tandemed LAI call	Look-Ahead Interflow DNIS information from the original LAI call.
No redirection at the sending switch	VDN name according to Override rules at the sending switch (active VDN).

DNIS information displayed for LAI scenarios (continued)

Scenario	Information displayed
Redirection at the sending switch (VDN in coverage path)	Original VDN name, or If multiple VDNs are accessed, the name of the VDN that was last accessed by a **`route-to`** command.
Sending switch sends a blank DNIS Name field (that is, a name is not assigned to the sending switch “called” VDN) or the trunk group is administered to not send the LAI name (see Information Forwarding on page 151).	Name associated with the receiving VDN. This name can be changed according to the rules of VDN Override at the receiving switch.

Note:

VDNs that map to vectors that place LAI calls must have their ISDN Calling Party Number (CPN) prefixes administered. If an ISDN CPN prefix is not administered, the assigned VDN name is not sent. Instead, a DNIS of all blank space characters is sent and displayed on the answering agent’s terminal.

Originator’s display

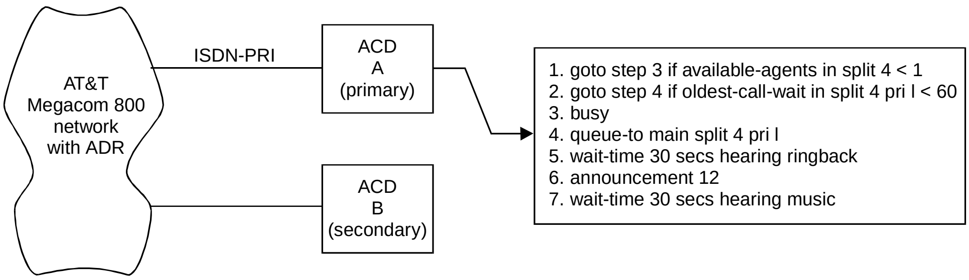
For internal calls, the originator’s display contains the same information as for Basic Call Vectoring. However, it is possible that the originator might receive undesirable display updates during LAI call attempts. To avoid this, ensure that the LAI calls are going out over trunk groups with the `Outgoing Display` field set to n. When the display field is set to no, internal callers who call that trunk group see the digits that they dialed on their display.

LAI with network ADR

Call Vectoring and LAI are compatible with and supplement the network services Alternate Destination Redirection (ADR) rerouting feature or equivalent service from other network providers. ADR uses ISDN-PRI connectivity with the switch in the same manner as LAI to allow the receiving system to indicate whether a call is to be accepted or rejected. The same type of vector that is used as a receiving ACD for LAI is used at the ADR-receiving ACD. If the call is accepted, it is connected to the system. If the call is rejected, the network routing number is translated to another number that routes the call to the alternate location within dialing-plan constraints. ADR allows for only one alternate location. LAI can be used at the alternate location to test other locations for less-busy conditions.

The following figure shows the configuration for a multilocation application.

ADR Example



The network requires ISDN-PRI connectivity to primary location A. Connection to secondary location B may or may not be ISDN-PRI. ADR attempts to route the call to location A over the ISDN-PRI link using a routing number that selects a VDN that is assigned to the receiving vector shown.

When the routing attempt is made, Call Vectoring starts processing the vector. The example then proceeds at location A as follows:

1. Step 1 checks for staffing of the ACD split, and branches to step 3 if it is not staffed.
2. If the ACD split is staffed, step 2 checks the oldest call waiting time in the split, and branches to step 4 if it is less than 60 seconds.
3. If the ACD split is unstaffed or if the oldest call waiting time is 60 seconds or more, step 3 rejects the call and returns a busy indication to the network.
4. If the oldest call waiting time is less than 60 seconds, step 4 accepts the call and queues it. ADR then connects the call through to the receiving system.
5. Steps 5 through 7 provide ringback, announcement, and music to the caller.

If the vector at location A rejects the call by sending a busy indication back to the network over the ISDN-PRI link, ADR reroutes the call to location B which must accept the call. If location B is closed or too busy to take the call, location B can use Call Vectoring and LAI to check other locations. If other locations exist and can take the call, location B can forward the call. If other locations do not exist or cannot take the call, location B can use Call Vectoring to route the call to location A. If location A is not open, location B can use Call Vectoring to provide an announcement or a busy tone to the caller.

Multi-site applications for Enhanced LAI

Enhanced LAI has two principal applications in a multi-site environment.

- It is possible to implement single-queue FIFO operation for any application. However, in many cases, Avaya recommends the use of BSR instead of LAI for maximum efficiency and flexibility. For more information, see Best Service Routing (BSR) on page 229.
- LAI can be used in combination with BSR for those switches in the network with extremely low call volumes.

  For more information about using BSR and LAI together, see Appendix E: Advanced multi-site routing on page 531

LAI considerations

The following are considerations for working with LAI:

- Never interflow to a remote vector that in turn might interflow back to the same local vector. This could cause a single call to use up all available trunks.
- Do not use the **`oldest-call-wait`** test condition in LAI vectors. OCW corresponds to the very next call to be answered and, as such, this test condition gives no information on the current state of call overload. For example, if OCW = 30 seconds, all we know from this is that the queue was overloaded 30 seconds ago. In place of **`oldest-call-wait`**, use the EWT conditional. For more information, see Expected Wait Time (EWT) on page 126.

- If an LAI call attempt is accepted by a step that contains a **`queue-to`**, **`check split`**, or **`route-to`** command, there is a small but finite interval during which the call could be answered by an agent at the sending switch before notification of acceptance is received by the sending switch. In this case, the caller is connected to the agent at the sending switch, while the agent at the receiving switch might receive a phantom call. For this reason, consider using a short **`wait-time`** or **`announcement`** step at the receiving switch to allow the call to be accepted and taken out of the queue at the sending switch. If call acceptance is to be based on available agents, use of a **`wait-time > 0`** seconds or an **`announcement`** is not recommended. A **`wait-time`** with 0 seconds of silence might be useful in this case.

  Note:

  For enhanced LAI operation, there are capabilities built into the feature to eliminate or reduce the occurrence of phantom calls. If phantom calls are a problem in an enhanced LAI operation, the `Interflow-Qpos EWT Threshold` field has been set too low.

- When an LAI call attempt is made, the TTR (if attached) is disconnected, and any dial-ahead digits are discarded. This implies that a subsequent **`collect digits`** command would require that the TTR be connected.
- Be sure that the feedback provided by the receiving switch after a successful LAI attempt is consistent with what the caller has already received.
- It is perfectly acceptable for a vector to route a call over an ISDN-PRI facility to a destination that is not a VDN. In this case, the sending switch treats the call as if it were a LAI call. Generic ISDN processing at the receiving switch causes the call to be accepted. The DNIS name is ignored.
- If a LAI call terminates to a VDN on a receiving switch where the LAI option is not enabled, intelligent interflow still results. However, any relevant DNIS information is ignored, and intelligent interflow to far-end switches is not possible.
- The LAI time-out in the sending switch occurs after 2 minutes.
- T-1 equipment might modify the ISDN D-channel that is used for LAI. If multiplexors are introduced into the ISDN-PRI circuit, bit compression and echo cancellation must be turned off for the D-channel.

Troubleshooting for LAI

The following are troubleshooting suggestions when working with LAI:

- If remote agents are experiencing a high volume of phantom calls, the `Interflow-Qpos EWT Threshold` may be set too low or too high.
- If remote agents are experiencing a delay between becoming available and receiving a call, the following may be the cause:
  - — The `Interflow-Qpos EWT Threshold` may be set too low.
  - — An insufficient number of LAI attempts have been made from the sending switch. In this case, change the **`interflow-qpos`** conditional at the sending switch. For example, change **`interflow-qpos=1`** to **`interflow-qpos <= 2`**.
  - — An insufficient number of tie trunks are available.
- If remote agents are receiving no calls, the maximum number of vector steps that are executed at the sending switch vector may have been reached before calls reached the head of the queue. In this case, rewrite the vector on the sending switch.

Chapter 12: Best Service Routing (BSR)

Best Service Routing (BSR) allows the switch to compare specified splits/skills, determine which will provide the best service to a call, and deliver the call to that resource. If no agents are currently available in that split/skill, the call is queued. To respond to changing conditions and operate more efficiently, BSR monitors the status of the specified resources and adjusts call processing appropriately.

BSR can be configured for either single-site or multi-site operation. Single-site BSR compares splits/skills on the switch where the BSR resides to find the best resource to service a call. Multi-site BSR extends this capability across a network of switches, comparing local splits/skills, remote splits/skills, or both, and routing calls to the resource that provides the best service.

This chapter includes the following topics:

Benefits of Best Service Routing

Both single-site and multi-site BSR intelligently compare specific resources to find the one that can best service a call. In addition, multi-site BSR makes it possible for you to integrate a network of contact centers for better load balancing and optimal agent utilization. Depending on your specific application, BSR can yield a variety of other benefits as shown in the following table.

Note:

If a contact center network is heavily overloaded and a significant number of calls are being blocked or abandoned, shorter wait times may not result when BSR is used. Rather than reducing wait times, any productivity gains will allow more calls to gain access to the network.

Best Service Routing benefits

You can benefit from…	As a result of…
Increased revenue	• Better agent utilization, thus allowing more calls to be handled with a given staff level. • Lower abandonment rates – By balancing the load between resources, BSR reduces extremes in wait times across local resources or across an entire network. • In contact centers with Expert Agent Selection, the ability to deliver calls to the best qualified or highest revenue generating agents.
Lower costs	• Better agent utilization. • Shorter trunk holding times. • Reductions of ineffective interflows. • Operation over ISDN-BRI trunks and public networks.

Best Service Routing benefits (continued)

You can benefit from…	As a result of…
Improved customer satisfaction	• Interflowing calls from centers with a surplus of calls to centers with a surplus of agents. You can achieve uniform service levels across your network. This means that all callers for a given application experience approximately equivalent waiting times. • Shorter wait times. • In contact centers with Expert Agent Selection, the ability to deliver calls to the best qualified or highest revenue generating agents. • Robust information forwarding capabilities. Multi-site BSR can forward original service requirements and any caller-entered digits with each call, and can use both QSIG and non-QSIG information transport methods over private or public networks.
Increased performance and more efficient trunk usage	• Less messaging and processing required per call than in traditional LAI scenarios. • Eliminates phantom calls to remote agents. • Intelligent interflows that only route calls to centers with available agents.
BSR's easy configuration	• Simple vector commands. You do not need to learn complex programming languages or design comparison steps. All that you have to do is list the local and remote resources to be considered for calls and instruct the switch to queue or deliver the call to the best resource on the list.
Improved agent productivity	• Increased efficiency. Improve your service without adding staff, or reduce staff while maintaining your current level of service. Network-wide load balancing means that agents at one location are less likely to sit idle while calls wait in queue at another location. • No call delivery delays. In contrast to approaches that queue calls at all remote centers simultaneously, with BSR there is no delay in delivering a call when an agent becomes available.

Best Service Routing benefits (continued)

You can benefit from…	As a result of…
Increased operating flexibility, easier staffing and scheduling	● Larger pool of agents available to take calls in a split/skill. Through its network-wide call distribution and information forwarding, BSR effectively converts distributed locations into a virtual contact center. Thus, staffing problems do not need to be solved on a center-by-center basis. BSR can automatically react to staff shortages at one center by routing more calls to other locations. ● Automatic management of sudden and unexpected increases in call volume. Large increases in call volume for a single split/skill can be distributed across other splits/skills. Spikes in call volume at a single contact center can be distributed across all contact centers, provided that sufficient trunk capacity is available between switches.
Improved service levels	● Lower average speed of answer (ASA).

Switch and network requirements for BSR

For single-site BSR applications, your switch must meet the requirements that are shown below. The requirements for ISDN trunks and LAI do not apply to single-site BSR applications.

To use multi-site BSR applications, all switches involved and the network connecting them must meet all of the requirements that are described in this section.

CAUTION:
To ensure that your network meets the requirements for BSR support, contact your Account Executive about BSR network certification.

Switch requirements

Your switch must meet the requirements shown in the following table to support BSR.

Requirements to use Best Service Routing

Form	Page	Field	Must be set to…
System-Parameters Customer-Options	2	ISDN-BRI Trunks[1]	Y
		ISDN-PRI Trunks[1] [2]	Y
	3	Vectoring (G3V4 Advanced Routing)	Y
		Vectoring (Best Service Routing)	Y
		Lookahead Interflow (LAI)[3]	Y
Feature-Related System Parameters	8	Adjunct CMS Release	R3V6 or higher, or left blank

1. Multi-site BSR operates over both BRI and PRI trunks. ISDN connectivity is only necessary if you want to use multi-site BSR, in which case one or both of these fields must be set to “Y.”

2. ATM trunking and IP trunking can be set up to emulate ISDN PRI. For information on setting this up, see the *Administration for Network Connectivity for Avaya MultiVantage Software*, 555-233-504, and *ATM Installation, Upgrades and Administration for Avaya DEFINITY Servers*, 555-233-124.

3. Look-Ahead Interflow is only necessary if you want to use multi-site BSR.

Tip:

If you begin using BSR and then turn it off, you can not set `Vectoring (Best Service Routing)` to **n** until you remove all BSR commands from vectors. If you are using multi-site BSR with Look-Ahead Interflow and want to turn LAI off, you can not set `Lookahead Interflow (LAI)` to **n** until you remove all **`consider location`**, **`reply-best`**, and **`interflow-qpos`** commands from vectors.

Network requirements

To support multi-site BSR, networks must meet both the criteria for LAI call control operation over switched networks (see Look-Ahead Interflow (LAI) on page 203) and the following criteria:

- The network must support end-to-end transport of codeset 0 user data, either as a User-to-User Information Element (UUI IE) or by QSIG Manufacturer Specific Information (MSI IE), in the ISDN SETUP and DISCONNECT messages. For more information, see Determining user information needs in Information Forwarding on page 151.
- With BSR poll calls, the information is forwarded back in the DISCONNECT message. In this case, the network must support forwarding of UUI in the first call clearing message, while the call is still in the call proceeding state, prior to the active state.
- Private networks can be configured for either QSIG (using MSI packaged in codeset 0 Facility IEs) or non-QSIG (using a codeset 0 UUI IE) transport. Currently, public networks do not support QSIG and user data can only be transported by the UUI IE when supported by the network. Future public network offerings may support QSIG, possibly by Virtual Private Network.
- The switch must support the ISDN country protocol.
- The network byte limit for the user data portion of the user information contents must be large enough to carry the data needed for the customer application.

Note:

Some public network providers may require service activation, fees for user information transport, or both.

BSR, LAI, enhanced information forwarding, and UCID have been tested with several major carriers. To find out if these capabilities work with your carrier, check with your account team for the most current information.

If testing has not been done to verify operation over the public networks that are involved with the preferred specific configuration, use of private ISDN trunking between the nodes should be assumed until successful testing is complete.

Terms to know

Understanding the BSR terms listed below will be helpful as you read through the material in this chapter. The following list contains terms pertaining to both single-site BSR and multi-site BSR.

adjusted EWT. Expected Wait Time plus a user adjustment set by a **`consider`** command.

agent selection method. The method that the switch uses to select an agent in a hunt group when more than one agent is available to receive the next call. Possible methods are:

- UCD-MIA
- UCD-LOA
- EAD-MIA
- EAD-LOA
- PAD

The agent selection method is a property of hunt groups and is set in the Group-Type field on the Hunt Group form.

To use any EAD available agent strategy, you must have Expert Agent Selection (EAS).

application – A general term for a system in any contact center that handles calls of a particular type. In relation to BSR, any specific implementation of multi-site BSR.

application plan – Used only in multi-site applications, the application plan identifies the remote switches that may be compared in consider series. The plan also specifies the information that is used to contact each switch and to interflow calls to it.

best – Includes the following conditions

- No agents available – When no agents are available in any of the specified splits/skills, the “best” resource is the one with the lowest adjusted EWT.
- Agent available in one resource – When an agent is available in one and only one of the splits/skills that are specified in a consider series, that agent is the “best” and the call is delivered to that agent. If the `BSR Available Agent Strategy` field is set to `1st-found`, BSR ignores all subsequent steps in the consider series. If any other available agent strategy is used, all remaining resources are still considered before the call is delivered.
- Agents available in two or more resources – When agents are available in two or more splits/skills, the “best” agent is the one that best meets the criteria that are specified in the `BSR Available Agent Strategy` field. For example, if the available agent strategy is UCD-MIA, the best agent out of those available will be the agent with the longest idle time.

Best Service Routing (BSR) – A feature that is based on call vectoring and routes ACD calls to the resource that is best able to service each call. BSR can be used on a single switch, or it can be used to integrate resources across a network of switches.

BSR available agent strategy – A field that appears on the VDN form when either version of BSR is enabled. The entry in this field is a property of the VDN and its assigned vector. Possible entries are:

- 1st-found
- UCD-MIA
- UCD-LOA
- EAD-MIA
- EAD-LOA
- PAD

When the VDN is the active VDN for a call, as determined by VDN Override, this field determines how BSR commands in the vector identify the best split/skill when several have available agents.

consider series – `consider` commands are typically written in a set of two or more. This set of **`consider`** commands is called a consider series. A consider series in a status poll vector might have just one consider step.

consider sequence – A consider sequence is a consider series plus a **`queue-to best, check-best, or reply-best`** step.

Expected Wait Time (EWT) – Expected Wait Time is an estimate of how long a call in the queue will have to wait before it is connected to an agent.

Intelligent polling – An automatic feature of BSR that significantly reduces the number of status polls that are executed. When a remote location cannot be the best resource at a given moment in time, the intelligent polling feature temporarily suppresses polls to that location.

interflow – The process of routing an incoming call to an external switch without answering it at the origin switch.

poll suppression – A component of BSR intelligent polling that eliminates wasteful polling of remote locations which have returned poor adjusted EWTs.

resources – An agent, split, skill, or location

status poll – A call that is placed by a **`consider location`** vector command to obtain status data from a remote location in a multi-site BSR application.

Single-site BSR

Single-site BSR is a simple, logical extension of call vectoring. Like any other vector, vectors with BSR commands are assigned to one or more VDNs. Using new vector commands and command elements, you tell the switch to compare, or “consider,” specific splits/skills for each call that is processed in that particular vector. Throughout the comparison, the switch can remember which resource is the best based on how you define “best.” BSR vectors can deliver a call to the first available agent found, or they can consider all of the specified resources and deliver the call to the best split/skill. If no agents are available in any split/skill, the call is queued to the split/skill with the shortest adjusted EWT.

Command set – single site BSR

The following table shows the forms, the vectors, and the vector commands and command elements that are used in single-site BSR. The following table shows the vector commands and command elements used in single-site BSR applications.

Vector commands and usage for single-site BSR

Commands and command elements		**Use this…**
Forms	Vector Directory Number form	To link a VDN to a BSR vector. To set the agent selection strategy that will be used for all calls to that VDN.
	Call Vector form	To confirm that BSR is administered. To write vectors that use BSR commands.

Vector commands and usage for single-site BSR (continued)

Commands and command elements		Use this…
Commands	**`consider split/skill`**	To obtain the Expected Wait Time or agent data that is needed to identify the best local resource. One **`consider`** step must be written for each split/skill that you want to check. Since the **`consider`** command is designed to compare two or more resources, **`consider`** commands are typically written in a series of two or more with the sequence terminating in a **`queue-to best`** vector step. This set of **`consider`** commands and a **`queue-to best`** step is called a consider sequence.
	`queue-to`	With the **`best`** keyword to queue calls to the best resource that is identified by the consider sequence.
	`check`	With the **`best`** keyword to queue calls to the best resource that is identified by the consider sequence if the resource meets certain conditions.
Key word	**`best`**	Use the **`best`** keyword in **`queue-to`**, **`check`**, and **`goto`** commands that refer to the resource that is identified as best by a series of **`consider`** steps
Conditional	**`wait-improved`**	To prevent calls from being queued to an additional split/skill when the reduction in Expected Wait Time is not enough to be useful. "Wait improved" means that a call's EWT must be improved by a specific amount, specified in seconds, over its current EWT or the switch does not queue the call to the additional split/skill.

Vector commands and usage for single-site BSR (continued)

Commands and command elements		Use this…
User adjustment	`adjust-by`	To specify your preferences for the splits/skills that might handle the calls for a particular application, reflecting factors such as agent expertise or reducing calls to a backup split/skill. When a vector considers a local resource you can make the selection of that split/skill less desirable. The higher the setting, the less chance that resource will be selected over another with a lower setting (for example, set to 30 makes that choice 30% less desirable). With EWT returned, the setting increases the returned expected wait time for comparison with other returned EWTs. As a result, this split/skill is less likely to service the call unless its EWT is significantly less than that of any other available split/skill. Optionally, the adjust-by setting applies in the available agent case. If you are using the UCD-MIA or EAD-MIA available agent strategy, the setting decreases the returned agent idle time, making the agent appear less idle (busier). If you are using the UCD-LOA or EAD-LOA available agent strategy, the setting increases the returned agent occupancy, making the agent appear more occupied (busier). In either case with EAD, the MIA or the LOA is used as a tie breaker if more than one site has an agent available with the same highest skill level.

How BSR determines the best resource

BSR determines the best resource to service a call by examining one or all of the following variables:

- The EWT of the resource
- Any user adjustments
- The availability of agents
- The selection strategy for the active VDN.

Note:

The BSR available agent strategy that applies to a given call is the strategy that is assigned to the active VDN for that call, as determined by VDN override.

Call surplus situations

Every BSR application compares a set of predetermined resources (splits/skills) and selects the “best” resource to service the call.

In a call surplus situation when no agents are available, the best resource is the split/skill with the lowest Expected Wait Time (EWT). For purposes of calculating the best resource in a call surplus situation, BSR allows you to adjust the EWT figure for any split/skill. The actual EWT for calls in queue is not changed. Only the figure used in the calculations performed by the BSR feature is changed. You do not have to enter adjustments, but the ability to adjust the EWT for splits/skills allows you to program preferences in vectors. Because of agent expertise, for example, or the availability or cost of tie trunks, you might prefer that some resources do not service a call unless doing so significantly decreases the time in queue for the call.

It is possible for you to make adjustments to agent availability using the **`consider`** step. For more information, see Agent selection adjustments on page 241.

Agent surplus situations

In an agent surplus situation when one or more agents are available to take incoming calls, BSR delivers a new call according to the `BSR Available Agent Strategy` that is specified on the VDN form. The “best” resource is the split/skill that meets the criteria that are defined by the strategy that was administered for that VDN. BSR can use any of the five strategies shown in the following table to select an agent when agents are available.

BSR available agent strategies

If BSR Available Agent Strategy is set to…	The call will be delivered to…
1st-found	The first available agent. BSR will not consider any other resources as soon as it finds an available agent.
ucd-mia	The resource with an agent who has been idle for the longest amount of time. BSR compares all of the splits/skills that are specified in the vector before delivering the call.
ead-mia	The resource with an agent who has the highest skill level that is relevant to the call and who has been idle the longest. BSR compares all of the splits/skills that are specified in the vector before delivering the call.

BSR available agent strategies (continued)

If BSR Available Agent Strategy is set to…	The call will be delivered to…
ucd-loa	The resource with the least-occupied agent. BSR compares all of the splits/skills that are specified in the vector before delivering the call.
ead-loa	The resource with an agent who has the highest skill level that is relevant to the call and who is the least occupied. BSR compares all of the splits/skills that are specified in the vector before delivering the call.

For more information on LOA, see *Avaya MultiVantage Call Center Software – Guide to ACD Call Center*, 555-230-716, or *Avaya Business Advocate User Guide*, 585-210-711. LOA is available with the Call Center Elite package.

When agents are available in one or more of the specified resources, BSR does not consider resources (local or remote) that return an EWT (call queue/call surplus situation) in selecting the best place to send the call.

Note:

The `BSR Available Agent Strategy` that is assigned to a VDN should match the agent selection method that is used in the splits/skills considered by a BSR application.

Agent selection adjustments

An option has been provided to have the BSR adjust-by value apply in the agent surplus (agents available) situation. This adjustment provides the ability to use the **`consider`** step adjustment value to prioritize (handicap) agent resources when agents are available.

When the adjustment is used, the **`consider`** step uses the following syntax:

`consider split/location adjust-by` ***`x`***

The switch applies the agent adjustment in the same manner as the calls in queue/call surplus (lowest EWT) situation.

To select an adjustment, think in terms of reducing the importance of a resource/site and in relative percentage — the higher the adjustment, the less desirable it is to pick that agent/site. So, if ***x*** = 30, then the agent/site is 30% less desirable.

The available agent adjustment applies to the UCD-MIA, UCD-LOA, EAD-MIA, and EAD-LOA call distribution methods. For the most idle agent distribution methods, the adjust-by lowers the idle time value returned by the agent/site. For the least occupied agent distribution methods, the adjust-by raises the returned occupancy level of the agent/site. In either case, with EAD, the MIA or LOA is used as a tie breaker if more than one site has an agent available with the same highest skill level.

The same adjust-by value in the **consider** step applies to both agent surplus and call surplus situations.

Example of basic single-site BSR

This example shows the simplest use of BSR. The central element of all single-site and multi-site BSR is a VDN/vector pair. The vector contains the commands that actually process the call, but the active VDN for the call contains information that is used by some vector steps. For single-site BSR, the active VDN for a call sets the available agent strategy that is used by the vector.

Single-site BSR example VDN Form

```
change vdn xxxxx                                                    page 1 of 2
                      VECTOR DIRECTORY NUMBER

                           Extension: 5000
                                Name: Single-site BSR
                       Vector Number: 234
                 Attendant Vectoring? n
                  Meet-me Conference? n
                  Allow VDN Override? n
                                 COR: 59
                                  TN: 1
                            Measured: internal
       Acceptable Service Level (sec): 20
        VDN of Origin Annc. Extension:
                           1st Skill:
                           2nd Skill:
                           3rd Skill:
```

```
change vdn xxxxx                                                   page 2 of 2
                      VECTOR DIRECTORY NUMBER

                          Audix Name:
                Messaging Server Name:
                   Return Destination:
               VDN Timed ACW Interval:
                     BSR Application:31
         BSR Available Agent Strategy: 1st-found
               Conference Access Code:
                Conference Controller:
         Display VDN for Route-To DAC?
```

In the example Vector Directory Number form shown above, the `BSR Available Agent Strategy` field is set to `1st-found`. If vector 234 uses BSR commands, as soon as a **`consider`** step locates a resource with an available agent any subsequent **`consider`** steps are skipped and the call is delivered to that resource. Resources that are specified in any subsequent **`consider`** commands are not checked. If no split has an available agent, the call is queued to the split with the lowest adjusted EWT.

If the `Allow VDN Override?` is set to n and a second VDN and vector are used to process this call, the 1st-found strategy specified in VDN 5000 will still be used.

In the preceding example, Vector Directory Number 5000 is associated with vector 234, which is shown below. In this example, vector 234 compares two splits. No adjustment is assigned to either resource, indicating that both splits are equally suited to service calls since neither is preferred to the other. In reality, such a vector would probably have additional steps after step 4, such as **`announcement`** or **`wait-time`** commands. These steps are omitted in this example for purposes of clarity.

Single-site BSR example vector

```
1. wait time 0 secs hearing ringback
2. consider split 1 pri l adjust-by 0
3. consider split 2 pri l adjust-by 0
4. queue-to best
```

Notice that the **`consider`** commands follow each other in unbroken sequence and that the **`queue-to best`** command immediately follows the last **`consider`** command. This structure is called a “consider series,” and it is recommended that you typically write such series in uninterrupted order. A few commands, such as the **`goto`** command, which cause little if any delay in the execution of the **`consider`** steps, may be used. In general, however, do not put other commands between **`consider`** steps, or between a **`consider`** step and a **`queue-to best`** step. Even if BSR still works in that situation, you might seriously impair the performance of the vector.

Consider commands collect and compare information. When a call is processed in the vector above, the first consider step collects and temporarily saves the following information about split 1:

- The fact that split 1 is a local split
- The queue priority that is specified in the **`consider`** step
- The user adjustment that is specified in the **`consider`** step
- The split’s
  - Split number
  - Expected Wait Time

If EWT=0, which indicates that one or more agents are available, the step also collects all of the agent information that might be needed by the BSR available agent strategy. This includes:

- Agent Idle Time (AIT)
- Agent Occupancy (AOC)
- The skill level of the agent in the split/skill who will receive the next call

In the example shown above, neither split has an available agent when the consider series executes. If one did, the call would be delivered to that split by the **`queue-to best`** step. Since there are no available agents in either split, the complete set of saved data now defines the “best” resource—for the moment. The second consider step collects the same data and compares it to the current “best” data. For this example, assume that the EWT for split 1 is 40 seconds and the EWT for split 2 is 20 seconds. When the second **`consider`** step executes, its data will replace the “best” data from step 1 because its adjusted EWT is lower. The “best” data is essentially a placeholder. When a **`queue-to best`** step executes, it reads the data that is saved as the “best” at that moment and queues the call to that split. In this case, the best data was collected from split 2, so the call is queued to split 2 at the specified priority.

Questions

What if there are available agents in both splits?

Since the `BSR Available Agent Strategy` in this example is 1st-found, the consider series will skip any **`consider`** steps after step 2 and the **`queue-to best`** step will deliver the call to split 1, which is the first split/skill with an available agent that is found by the vector.

In any BSR vector, the order of the **`consider`** steps should reflect your preferences for the resources to be considered. Put the step that considers the most preferred split/skill first, the step for your second preference second, and so forth in the consider series.

What if there are several available agents in split 1? Which agent receives the call?

When more than one agent is available in a split, the BSR **`consider`** command collects agent data only for the agent who will receive the next call to that split. This agent is identified according to the agent selection method that is specified in the `Group-Type` field on the Hunt Group form.

Note:

For greatest efficiency, the agent selection method used in the splits/skills considered by a BSR vector should match the `BSR Available Agent Strategy` that is assigned to the active VDN.

User adjustments in single-site BSR

You may have preferences as to which splits/skills should answer certain types of calls. In both single-site BSR and multi-site BSR, the **`adjust-by`** portion of the **`consider`** command makes it possible for you to program these preferences into your vectors.

You can assign a value of 0 to 100 in user adjustments. The units of this value are supplied by the switch depending on the conditions whenever that **`consider`** step executes. For example, in the command **`consider split 1 pri h adjust-by 20`**, the switch interprets **`adjust-by 20`** to mean “add 20% to the EWT, but add at least 20 seconds.”

Note:

If the user adjustment were defined as a number of seconds, BSR would not be efficient when EWT was high. If the user adjustment were defined as a percentage, BSR would not be efficient when EWT was low. Such efficiencies, while always important, become critical in multi-site BSR applications where issues of trunk cost and capacity are involved.

For Expected Wait Times of 1 to 100 seconds, an adjustment of 20 will therefore add 20 seconds. Above 100 seconds, the same adjustment will add 20% to the EWT for the split/skill that is specified in the **`consider`** step. The following table shows the results of applying a constant adjustment to a range of Expected Wait Times.

User adjustments in BSR

EWT of resource (seconds)	User adjustment	Adjustment applied by the switch (seconds)	Adjusted EWT used to select resource
10	20	20	30
60		20	80
120		24	144
300		60	360

Example of single-site BSR with adjustments

The following example shows a more complex implementation of single-site BSR. Four skills in an Expert Agent Selection environment are compared. The Expected Wait Time (EWT) for some skills is adjusted to reflect the administrator's preferences

Single-site BSR example VDN form

```
change vdn xxxxx                                                  page 1 of 2
                      VECTOR DIRECTORY NUMBER

                            Extension: 5001
                                 Name: Single-site BSR
                        Vector Number: 11
                  Attendant Vectoring? n
                   Meet-me Conference? n
                   Allow VDN Override? n
                                  COR: 59
                                   TN: 1
                             Measured: internal
       Acceptable Service Level (sec): 20
        VDN of Origin Annc. Extension: 501
                            1st Skill:
                            2nd Skill:
                            3rd Skill:
```

```
change vdn xxxxx                                                 page 2 of 2
                      VECTOR DIRECTORY NUMBER

                           Audix Name:
                 Messaging Server Name:
                    Return Destination:
                VDN Timed ACW Interval:
                       BSR Application:19
         BSR Available Agent Strategy: EAD-MIA
            Observe on Agenst Answer?:
                Conference Access Code:
                 Conference Controller:
         Display VDN for Route-To DAC?
```

In the example shown above, the `BSR Available Agent Strategy` field is set to `EAD-MIA`. If vector 11 uses BSR commands, calls are not automatically delivered to the first resource with an available agent that is found. All **`consider`** steps in vector 11 are executed and one of the following things happens:

If …	Then…
No skill has an available agent	The call queues to the skill with the lowest adjusted EWT.
Only one skill has an available agent	The call is delivered to that skill.
Two or more skills have available agents	The call is delivered to the skill with the most expert agent.
Two or more skills have available agents with the same skill level	The call is delivered to whichever of these agents has been idle the longest.

Also note that `Allow VDN Override?` is set to `n`. If a second VDN and vector are used to process this call, the EAD-MIA strategy that is specified in VDN 5001 is used. If `Allow VDN Override?` is set to `y` and vector 11 routes some calls to another VDN, the subsequent VDN's available agent strategy governs the operation of **`consider`** steps in its vector.

The following example vector 11, which compares four skills.

Single-site BSR example vector

```
1. wait-time  0 secs hearing ringback
2. consider skill  1 pri l adjust-by 0
3. consider skill  2 pri l adjust-by 30
4. consider skill 11 pri l adjust-by 30
5. consider skill 12 pri l adjust-by 30
6. queue-to best
7. wait-time 10 secs hearing ringback
8. announcement 1001
9. wait-time 30 secs hearing music
10. goto step 8 unconditionally
```

For this example, assume that the Expected Wait Times of the four skills are 95, 60, 180, and 50 seconds, respectively. Notice that all **`consider`** steps except the first adjust the EWT returned by the specified skill. Skill 1 is the preferred skill to handle calls to VDN 5001, so its EWT is not adjusted. Skills 2, 11, and 12 can handle this call type, but they are not preferred. The adjustment of 30 means that, in call surplus situations, these skills will not handle calls to VDN 5001 unless their EWT is at least 30 seconds better than the EWT in skill 1.

The following table shows the adjustments that would be applied to each skill given its EWT and the user adjustment specified in the **consider** step. The last column shows the adjusted EWT the switch will use to select a skill for the call.

User Adjustments

Skill number	User adjustment in the consider step	Actual EWT (seconds)	Adjustment applied by the switch (seconds)	Adjusted EWT used in BSR calculations (seconds)
1	0	95	0	95
2	30	60	30	90
11	30	180	54	234
12	30	50	30	80

Since the available agent strategy is not 1st-found, all four **consider** steps are executed each time that the vector processes a call. In this example, there are no available agents in any of the skills. In fact, EWT is high enough in the first three skills for the switch to queue the call to skill 12.

When the **queue-to-best** step executes, the data in the best data placeholder is the data from skill 12 and so the call is queued to that skill. From this point on, if the call is not answered during the execution of step 7, a common vector loop regularly repeats an announcement for the caller while he or she waits in the queue.

For DEFINITY software version 9.1 or later, user adjustments also apply to available agent situations (with a strategy other than first found) in a manner that is similar to EWT. For more information, see the “Best Service Routing” section in: *Avaya MultiVantage Call Center Software – Guide to ACD Call Center*, 555-230-716.

Questions

What if there is an available agent in one skill? Will user adjustments be applied?

Since the `BSR Available Agent Strategy` in this example is EAD-MIA, the entire consider series will always be executed to check all of the skills for available agents. If only one skill has available agents, the call is delivered to that skill and user adjustments are not applied.

What if there are available agents in two skills? Which skill gets the call? Will user adjustments be applied?

Since the `BSR Available Agent Strategy` for VDN 5001 (the active VDN) is EAD-MIA, the call is delivered to the skill with the most expert agent. If there are available agents in both skills with the same skill level, their user adjusted idle times are compared and the call goes to the skill with the agent who has the longest adjusted idle time.

If a split/skill has more than one available agent, remember that it is the split/skill's agent selection method that determines which agent's data is used in BSR selection of the best resource.

What if no agents are staffed in a skill? Will the switch recognize this?

Yes. Under any of the following conditions, the EWT returned from a split/skill is infinite:

- No agents logged in
- No queue slots available
- All agents in AUX work mode

The switch logs a vector event and goes to the next vector step without changing the data in the best placeholder. A resource with an infinite EWT is never selected as the best resource.

Can VDN skills be used in consider steps?

Yes. For example, **`consider skill 1st [2nd, 3rd] pri m adjust-by 0`** will collect data on the 1st [2nd, 3rd] skill, as defined for the active VDN.

Planning and administering single-site BSR

This section presents information that is specific to BSR. Follow existing procedures to add or change other properties of VDNs and vectors that are not discussed in this section.

First, confirm that your switch meets the requirements for single-site BSR if you haven't already done so. See Switch and network requirements for BSR for a listing of requirements.

Planning

To work more efficiently, you may want to record goals, VDN extensions, vector numbers, and other information on paper before you begin your administration session. To do this, complete the following:

1. Select the group of callers for which you want to use single-site BSR, and identify the VDNs and vectors that support this group.
2. Define your goals. For example, your goals in using BSR might be faster average speed of answer, or better service by routing calls to the most qualified agents.

   Different VDNs or vectors may have different goals.
3. Decide which agent selection strategy that you will assign to each VDN in order to best achieve the goals that are relevant to that VDN.
4. Decide whether you will allow VDN Override for each of the VDNs that are identified.

Administration

Use this procedure to administer single-site BSR, complete the following:

1. To go to the Vector Directory Number form for the first VDN you identified in step 1 of Planning, type **`add vdn`** ***xxxxx*** or **`change vdn`** ***xxxxx*** at the command line prompt and press Enter, where ***xxxxx*** is a valid VDN extension as defined in the system dial plan.
2. In the `Allow VDN Override`? field, enter **y** or **n**. If the call is directed to another VDN during vector processing:
   - — **y** allows the settings on the subsequent VDN, including its `BSR Available Agent Strategy`, to replace the settings on this VDN.
   - — **n** allows the settings on this VDN, including its `BSR Available Agent Strategy`, to replace, or override, the settings on the subsequent VDN.

3. In the `BSR Available Agent Strategy` field, enter the identifier for the agent selection method that you want this VDN to use.

   When this VDN is the active VDN for a vector that uses BSR, the available agent strategy determines how calls are directed when one or more of the specified resources have available agents. If there is only one split/skill with available agents, calls are delivered to that resource.

If you enter…	Consider series in vectors will select the resource with…
1st-found	The first available agent. BSR does not consider any other resources as soon as it finds an available agent.
ucd-mia	The agent who has been idle the longest. BSR will compare all of the splits/skills that are specified in the vector before delivering the call.
ead-mia	The agent with the highest skill level who has been idle the longest. BSR compares all of the splits/skills that are specified in the vector before delivering the call.
ucd-loa	The least-occupied agent. BSR compares all of the splits/skills that are specified in the vector before delivering the call.
ead-loa	The agent with the highest skill level who is the least occupied. BSR compares all of the splits/skills that are specified in the vector before delivering the call.
pad	The available agent with the lowest ratio of adjusted work time and target allocation for the skill.

4. Press **Enter** to save your changes.

You are now ready to write or modify the vector that is assigned to this VDN. For tips on using BSR commands in vectors, see Tips for writing BSR vectors on page 280.

Troubleshooting for single-site BSR

You should regularly execute a **`display events`** command for the appropriate vectors, especially if you have just implemented a new BSR application. Vector events will identify and indicate the source of common malfunctions and administration errors.

For a list of BSR vector events and definitions, see Troubleshooting vectors on page 493.

Note:

Only the most recent events are displayed when a **`display events`** command is executed. For this reason, you should periodically display vector events to help quickly identify problems.

To verify that your BSR vectors are operating as intended, use a **`list trace vdn`** or **`list trace vec`** command to observe processing of an individual call. See Clearing events on page 529 for more information.

Multi-site BSR

Multi-site BSR extends all of the capabilities of single-site BSR across a network of switches. Multi-site BSR can compare local splits/skills, remote splits/skills, or both, and route calls to the resource that provides the best service. In addition, multi-site BSR has special features that work to ensure efficient use of processor power and network resources in your BSR applications.

Throughout the rest of this chapter, the words “local,” “origin,” and “remote” are used to label different switches in multi-site applications. These words may seem to suggest that only one switch (the “local” or “origin” switch) in a network is receiving calls, polling other (“remote”) switches, and interflow calls. While such a centralized system may sometimes be useful, in most networks with BSR every switch in the network is able to interflow calls to other switches and receive interflowed calls from other switches. For clarity in the following discussions, “local” or “origin” means a switch that is considering whether to interflow a call. “Remote” means any switch that may be polled by this first switch and thus might receive the interflowed call. More generally, these terms are relative to the BSR applications that you design. In terms of a given application, the “local” or “origin” switch is the switch on which the Application Plan form for this application resides, and the “remote” switches are the switches that are identified at the locations listed on the form. See Multi-site BSR applications for an explanation of multi-site BSR applications.

When each switch in a network may interflow calls to other switches and receive interflows, this is called a “distributed” system. A “centralized” system, by contrast, is one in which all calls are initially delivered to a single contact center (the “hub”) and distributed from this site to queues at remote switches. A centralized system requires greater inter-switch trunking, since a greater percentage of calls need to be redirected. However, it may be an appropriate configuration if your organization has a significant investment in VRU and CTI technology at the hub.

Note:

The following material on multi-site BSR assumes that you already understand the operation of single-site BSR.

Multi-site BSR command set

The following table shows the forms, the vectors, and the special vector commands and command elements that you use to administer multi-site BSR applications. The table also briefly describes the purpose of each component. Since all of the command elements of single-site BSR can be used in multi-site applications, they are included in the table for convenience.Vector commands for multi-site BSR.

Vector commands and usage for multi-site BSR

Forms	
Best Service Routing Application Plan form	• To define the group of remote sites that will be polled by a specific application. • To assign a unique name and number to each application. • To assign routing numbers for the status poll and interflow VDNs.
Vector Directory Number form	• To link a VDN to a BSR application by its application number. • To link the VDN to a BSR vector. • To set the agent selection strategy that will be used for all calls to that VDN.
Call Vector form	• To confirm that BSR is administered and to program the vector steps for BSR.
ISDN Trunk forms	• To tell the switch whether to forward user information by Shared UUI or QSIG MSI.
List Best Service Routing Applications form	• To display a list of all the BSR applications by name and number.
System Capacity	• To monitor the number of BSR application-location pairs that are assigned in your system.
VDNs and Vectors	
Primary VDN (the active VDN for the call at the origin, as determined by VDN override)	• To define the application plan and available agent strategy that are used by the vector that is assigned to this VDN.
Primary vector	• To control call processing at the original switch and compare local and remote resources.
Status poll VDN/vector	• To respond to status poll calls from another switch. The status poll vector considers a set of local splits/skills and returns data on the best resource to the original switch.
Interflow VDN/vector	• To accept BSR calls from another switch and queue them to the best of the local resources considered.

Vector commands and usage for multi-site BSR (continued)

Commands	
`consider split/skill`	• To obtain the Expected Wait Time or agent data that is needed to identify the best local resource. One **`consider`** step must be written for each split/skill that you want to check. Since the **`consider`** command is designed to compare two or more resources, **`consider`** commands are typically written in a series of two or more with the sequence terminating in a **`queue-to best`** vector step. This set of **`consider`** commands and a **`queue-to best`** step is called a consider sequence.
`consider location`	• To obtain the Expected Wait Time or agent data that is needed to identify the best resource at a remote switch. One **`consider`** step must be written for each location that you want to check. Routing information is obtained from the BSR Application plan for the active VDN.
`reply-best`	• To return data to another switch in response to a status poll
`queue-to`	• With the **`best`** keyword to queue calls to the best resource that is identified by the consider sequence.
`check`	• With the **`best`** keyword to queue calls to the best resource that is identified by the consider sequence if the resource meets certain conditions.
Key word	
`best`	• In **`queue-to`**, **`check`**, and **`goto`** commands that refer to the resource identified as best by a series of **`consider`** steps
Conditional	
`wait-improved`	• To prevent calls from being queued to an additional split/skill—local or remote—when the reduction in Expected Wait Time is not enough to be useful. “Wait improved” means that a call’s EWT must be improved by a specific amount, which is a figure that you specify in seconds, over its current EWT or the switch will not queue it to the additional split/skill.

Vector commands and usage for multi-site BSR (continued)

User adjustment	
adjust-by	• To control long-distance costs and limit trunk usage, reflecting factors such as availability of the trunks or agent expertise at remote locations. When a vector polls a local or remote resource, you can make the selection of that site less desirable. The higher the setting, the less chance that resource will be selected over another with a lower setting. With EWT returned, the setting increases the returned expected wait time for comparison with other returned EWTs. Optionally, the adjust-by setting applies in the available agent case. If you are using the UCD-MIA or EAD-MIA available agent strategy, the setting decreases the returned agent idle time, making the agent appear less idle (busier). If you are using the UCD-LOA or EAD-LOA available agent strategy, the setting increases the returned agent occupancy, making the agent appear more occupied (busier). In either case with EAD, the MIA or the LOA is used as a tie breaker if more than one site has an agent available with the same highest skill level.

Multi-site BSR applications

You can implement BSR at a single location by using the BSR commands in vectors. Using BSR across a network is more complex and requires additional administration.

Since a series of **consider location** steps in a multi-site BSR vector contacts one or more remote locations, you need to define these locations, tell the switch how to contact each one, and set up VDNs and vectors to handle communications between the origin switch and the remote (or receiving) switches. The BSR application should support some larger application in your contact center that handles calls of a particular type.

Note:

Any mixture of split/skill numbers, VDN numbers, and vector numbers can be used to support a single customer application or call type across a network. For clarity and simplicity, it is recommended that the BSR Application Plan number and the location numbers for a given application be the same on all switches.

You also need to set up ISDN trunk groups, set the parameters for information forwarding (UUI Transport), and administer numbering plans and AAR/ARS tables.

Multi-site BSR starts with the active VDN for a call, as determined by VDN override. If you want any specific VDN/vector pair to interflow calls using multi-site BSR, you must create a specific application for it. A multi-site application must contain the elements shown in the following table.

The required elements of a multi-site BSR application

A BSR application consists of…	Which serves this purpose…
The Primary VDN	The Primary VDN is the active VDN for a call at the origin switch, as defined by VDN override. Therefore, the Primary VDN in a BSR application does not have to be the VDN that originally received the incoming call. The primary VDN links its assigned vector to a BSR application plan and sets the `BSR Available Agent Strategy`.
The Primary vector that handles the incoming call on the origin switch	The Primary vector contacts the specified remote switches, collects information, compares the information, and delivers or queues the call to the resource that is likely to provide the best service.
An application plan	The application plan identifies the remote switches that you can compare and specifies the information that will be used to contact each switch and to route calls to it.
Two VDN/vector pairs on each remote switch: ● Status poll VDN/vector ● Interflow VDN/vector	**Status poll VDN/vector** The status poll vector compares splits at its location and replies to the origin switch with information on the best of these splits. Each remote switch in a given application must have a dedicated status poll VDN/vector.
	Interflow VDN/vector When a given remote switch is the best available, the origin switch interflows the call to this VDN/vector on the remote switch. Each remote switch in a given application has to have a dedicated interflow VDN/ vector. The steps in this vector deliver or queue the call, as appropriate, to the best resource that is found by the status poll vector.

To create a multi-site BSR application, you start by creating an application plan on the origin switch.

Note:

Remember that the terms “local,” “origin,” and “remote” are relative terms. In most networks that use multi-site BSR, every switch can interflow calls to other switches and receive interflowed calls from other switches. Therefore, every switch in the network may have all the elements described above. For clarity in the following discussions, “local” or “origin” means a switch that is considering or might consider whether to interflow a call. “Remote” means any switch that is polled or might be polled by this first switch.

Application plans

The application plan identifies the remote switches that you can compare and specifies the information that is used to contact each switch and to route calls to it.

The plan for each application is identified by the application number and a name. It specifies the remote switches that might be polled by the application and identifies each with a number called the location number. The plan also specifies the numbers for the status poll and interflow VDNs for each remote switch. Whatever you would dial to reach these VDNs is what should be entered in these fields: full length numbers as well as AAR, ARS, UDP, or public network numbers will work.

You create application plans on the Best Service Routing Application form. A plan for an application with three remote switches might look like the following example.

Sample multi-site BSR Application Plan

```
BEST SERVICE ROUTING APPLICATION PLAN

 Number: 15   Name: Customer Service  Maximum Suppression Time: 60   Lock? y

Num  Location Name  Switch Node                        Status Poll  VDN Interflow VDN  Net Redir?
1    New Jersey        320                             84015           84115               n
2    Denver            18                              913031234015    913031234115        n
4    New York          12345                           912121234015    912121234115        n
___ ______________   ________                          ____________    ____________        n
___ ______________   ________                          ____________    ____________        n
___ ______________   ________                          ____________    ____________        n
___ ______________   ________                          ____________    ____________        n
___ ______________   ________                          ____________    ____________        n
___ ______________   ________                          ____________    ____________        n
___ ______________   ________                          ____________    ____________        n
___ ______________   ________                          ____________    ____________        n
___ ______________   ________                          ____________    ____________        n
___ ______________   ________                          ____________    ____________        n
___ ______________   ________                          ____________    ____________        n
```

Depending on your switch software release and platform, the maximum number of application plans is either 255 or 512, and the maximum number of application-location pairs is either 1000 or 2,560. For more information, see *System Capacities Table for Avaya MultiVantage on Definity Servers*, 555-233-605. To access the document online, go to http://avayadocs.com, select **Document Search**, and search for document number 555-233-605.

By entering the application number from this plan on a VDN form, you can link a given VDN on your local switch to this list of locations. This VDN becomes the primary VDN for the application. For example, if the primary vector contains instructions to consider locations 1 and 2, the switch places a status poll call to the status poll VDN at the New Jersey and Denver switches and compares the results. If location 2 is better than either location 1 or any splits that are considered on the originating switch, the call will be interflowed to the interflow VDN that is specified in the plan for location 2.

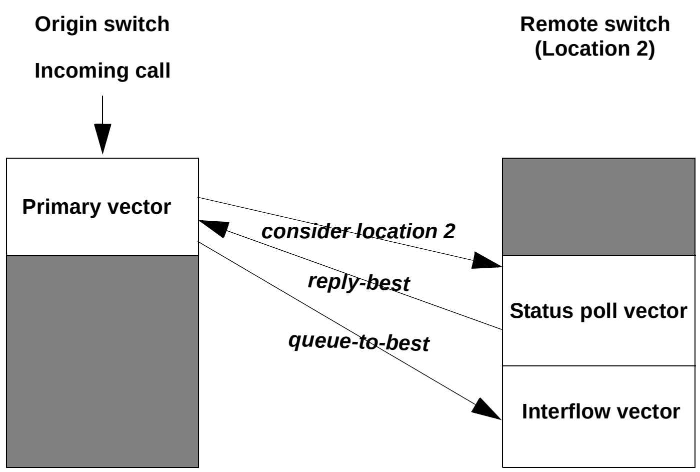
Example of multi-site BSR with two switches

To see how the basic elements of multi-site BSR work, consider a simple application in a two-switch network. Multi-site BSR compares local and remote splits/skills and queues calls to the resource that provides the best service. Remember that each BSR application has two main parts:

- An application plan. This plan identifies the remote switches that you want to compare.
- A set of three VDN/vector pairs:
  - — The primary VDN/vector. This vector on the origin switch contacts the specified remote switches, collects information, compares the information, and routes the call to the switch that is likely to provide the best service.
  - — The status poll VDN/vector. The status poll vector on the remote switch compares resources on that switch and replies to the origin switch with information on the best of these. Each remote switch in a given application must have a dedicated status poll vector.
  - — The interflow VDN/vector. When a given remote switch is the best available, the origin switch interflows the call to this vector on the remote switch. Each remote switch in a given application has to have a dedicated interflow vector.

The general operational scheme for multi-site BSR is shown in the following figure.

BSR example of origin and remote switches



The following example shows the primary VDN using a multi-site BSR application.

BSR example primary VDN

```
change vdn xxxxx                                                  page 1 of 2
                      VECTOR DIRECTORY NUMBER

                                Extension: 52222
                                     Name: Multi-site BSR
                            Vector Number: 222
                      Attendant Vectoring? n
                       Meet-me Conference? n
                       Allow VDN Override? n
                                      COR: 59
                                       TN: 1
                                 Measured: internal
           Acceptable Service Level (sec): 20
            VDN of Origin Annc. Extension:
                                1st Skill:
                                2nd Skill:
                                3rd Skill:
```

```
change vdn xxxxx                                                 page 2 of 2
                      VECTOR DIRECTORY NUMBER

                               Audix Name:
                     Messaging Server Name:
                        Return Destination:
                    VDN Timed ACW Interval:
                           BSR Application:15
             BSR Available Agent Strategy: UCD-MIA
                 Observe on Agent Answer?:n
                    Conference Access Code:
                     Conference Controller:
             Display VDN for Route-To DAC?
```

In the example shown above for VDN 52222, the entry in the `BSR Application` field links this VDN to BSR Application Plan 15. Also note the **UCD-MIA** entry in the `BSR Available Agent Strategy` field. If vector 222 uses BSR commands, calls are not automatically delivered to the first resource found with an available agent. All **consider** steps in vector 222 are executed, and one of the following things happens:

If:	Then:
There is no available agent in the local or the remote splits	The call will be queued to the split with the lowest adjusted EWT.

If:	Then:
Only one split has an available agent	The call will be delivered to that split.
Two or more splits have available agents	The call will be delivered to the split with the most idle agent.

Also note that `Allow VDN Override?` is set to n. If a second VDN and vector are used to process this call, the UCD-MIA strategy and the application plan that are specified in VDN 52222 are used.

Application plan 15 (which is shown in Sample multi-site BSR Application Plan on page 258) identifies the remote switch and provides the digit strings to dial into the VDNs for both the status poll vector and the interflow vector.

Primary Vector

When a call arrives at the origin switch, it is processed by the primary vector. This vector begins the BSR process by considering the resources that are specified. The following example shows what the primary vector might look.

BSR example of primary vector on origin switch

```
1.   wait time 0 secs hearing ringback
2.   consider split1 pri m    adjust-by 0
3.   consider location 2      adjust-by 30
4.   queue-to-best
```

In this example, the **`consider`** commands in steps 2 and 3 collect information to compare local split 1 with one or more splits at location 2. (Location 2 is the Denver switch identified on the BSR Application Plan form.) Step 4 queues the call to the best split that is found. As in single-site BSR, the **`adjust-by`** portion of the **`consider`** command allows you to set preferences for each resource, whether the resource is a remote location or a split/skill on the origin switch. In multi-site BSR, this user adjustment enables you to control the frequency of interflows by adjusting the EWT that is returned by a particular resource on a remote switch. In this example, the switch administrator has chosen to adjust the EWT value for location 2 by 30.

Status poll vector

To collect information from the remote switch, the command **`consider location 2 adjust-by 30`** in the primary vector places an ISDN call, known as a status poll, to the status poll vector on the switch at location 2. The following example shows what the status poll vector on the remote switch might look.

BSR example of status poll vector on remote switch

```
1.   consider split2 pri m              adjust-by 0
2.   consider split 11pri m             adjust-by 0
3.   reply-best
```

The status poll only obtains information and returns it to the origin switch; the call is not connected to the status poll VDN.

This vector compares splits 2 and 11, identifies the better of the two, and sends this information back to switch 1 with the **`reply-best`** command. Notice that the **`adjust-by`** command could be used on the remote switch to adjust the EWT that is returned by either of the splits. When EWT adjustments are applied at both the origin and remote switches, the two adjustments are added at the origin switch. See User adjustments in multi-site BSR on page 265 for more detail on user adjustments in multi-site applications.

The **`consider`** command is ISDN neutral and does not return answer supervision. The status poll call is dropped when the **`reply-best`** step executes, but the ISDN DISCONNECT message that is returned to switch 1 contains the information from the best split considered at location 2. Once the remote switch returns the necessary information, the consider series in the primary vector on switch 1 can continue at the next vector step.

CAUTION:

It is recommended that status poll vectors not be used to poll other switches. Status poll vectors should only consider resources on the switch where the vector resides. Status poll vectors must always end with a reply-best step. A busy or disconnect should never be used.

Note:

Multi-site BSR includes mechanisms that automatically limit the number of status poll calls that are placed over the network when such calls are unlikely to yield better service for the caller. For a detailed explanation of these mechanisms, see Appendix E: Advanced multi-site routing on page 531.

Interflow Vector

In this example, assume that no agents are available and that split 11 (location 2) has the lowest adjusted EWT. The **`queue-to best`** command in the primary vector will interflow the call to the interflow vector at location 2. The following example shows what the interflow vector looks like.

BSR example of interflow vector on remote switch

```
1.   consider split2          pri m              adjust-by 0
2.   consider split 11        pri m              adjust-by 0
3.   queue-to best
```

The interflow vector reconsiders the status of both splits to get the most current information and queues or delivers the call to the best split. Notice that the consider sequences in the interflow vector and the status poll vector are identical aside from their last step. When a call is interflowed, it is removed from any queues at the origin switch and any audible feedback at the origin switch is terminated.

CAUTION:

BSR will not operate correctly unless the consider series in the status poll vector and the interflow vector use the same splits/skills with the same queue priorities.

Questions

What happens to the call if the interflow attempt fails?

If the interflow attempt fails, for example, because there are no available trunks, the call is queued to the best local split. The call is not disconnected. The call is not dropped from vector processing on the origin switch. For the call to be queued to a local split, however, that split must have been the “best” resource at some previous point in the consider series. In writing primary vectors, always consider local splits/skills before considering remote resources.

I can adjust the EWT returned by a split/skill when no agents are available. When agents are available in two or more splits/skills, can I adjust Agent Idle Time (AIT) that is returned by a resource? Can I adjust the agent skill level returned by a resource?

No. EWT for a resource is the only data that BSR lets you adjust.

BSR available agent strategies

In multi-site BSR applications, the 1st-found available agent strategy results in fewer interflows and thus minimizes the load on interswitch trunking. The switch also has less processing to perform for each call in BSR vectors, since it may not need to compare as many resources to identify the best. If processing power and tie trunk capacity are issues in your multi-site applications, you may want to use the 1st-found strategy.

The other strategies typically result in a much greater percentage of calls being interflowed, thus optimizing load balancing across locations. For a strategy that greatly increases agent fairness across the network while limiting the number of trunks used, see Example of multi-site BSR with limited trunking on page 266.

More on status poll and interflow vectors

The following points are important to consider when you write status poll and interflow vectors.

- Since status poll vectors do not return answer supervision, call charges are not normally incurred for the status poll portion of the call flow.
- When a **`consider location`** step performs a status poll, it also checks for the availability of a B-channel. If no B-channel is available, the remote resource is never considered the best since the call cannot be redirected to it.
- If only one split/skill on a remote switch can service the call type that is handled in a BSR application, you do not need to write a consider series in the interflow vector. You can just queue the call to the appropriate resource.
- If status poll and interflow vectors consider more than one split/skill, the VDNs for these vectors must be administered with the appropriate BSR available agent strategy.

User adjustments in multi-site BSR

User adjustments are especially important in multi-site applications, where unnecessary interflows may be costly and use trunk capacity inefficiently.

User adjustments in multi-site applications function in the same way they do in single-site BSR with one important difference: user adjustments may be applied at the remote switches in an application as well as at the origin switch. Since a status poll vector uses **`consider`** steps to evaluate resources on the switch where it resides, the **`adjust-by`** portion of each **`consider`** command allows the administrator at each switch to set preferences for the splits/skills at that switch. In BSR applications, any such adjustment for a split/skill is considered by the status poll vector in selecting the best resource on its switch. The adjustment is then returned to the origin switch along with the other data for that resource. When the switch receives this adjustment from the remote switch, it adds it to any adjustment that was assigned to that location in the **`consider location`** step. The following example assumes, of course, that no agents become available during the time these vectors are processing the call.

The following example shows a primary vector that considers one remote location, to which it assigns an adjustment of 30.

Vector with consider step for one location

```
1.   wait time 0 secs hearing ringback
2.   consider split           pri m     adjust-by  0
3.   consider location 2                adjust-by 30
4.   queue-to-best
```

The following example shows the status poll vector at location 2.

Status poll vector

```
1.   consider split2 pri m              adjust-by 0
2.   consider split 11pri m             adjust-by 20
3.   reply-best
```

Consider split/skill commands in status poll vectors work just like they do in single-site BSR vectors. The user adjustments are applied to a single split/skill and not to the entire location. In this case, the two splits are assigned different adjustments. Say that split 11, despite having the larger adjustment, returns the lower adjusted EWT for a call. The **`reply-best`** command in step 3 returns the user adjustment of 20 to the primary vector on the origin switch, along with the rest of the data for split 11.

In saving the data that is returned by location 2, the origin switch adds the remote adjustment of 20 to the adjustment of 30 that is specified in step 3 of the primary vector. As a result, the call will not interflow to location 2 in this example unless the EWT for location 2 is more than 50 seconds better than the EWT in split 1 on the origin switch.

Example of multi-site BSR with limited trunking

Multi-site BSR applications must balance improvements in wait times and agent utilization with the cost of interflows and the availability of inter-switch trunking for status polls and interflows. The following example shows an application that is recommended for balancing agent workload across the network while still limiting tie trunk usage.

BSR example of Application Plan

```
BEST SERVICE ROUTING APPLICATION PLAN

 Number: 10   Name: International       Maximum Suppression Time: 60   Lock? y

Num  Location Name  Switch Node                        Status Poll  VDN Interflow VDN  Net Redir?

1    Kansas City      1111                             919131234015    919131234115          n
2    New York         1112                             912121234015    912121234115          n
3    Montreal         1113                             915141234015    915141234115          n
3    London           1114                             90114411234015  90114411234115        n
___ ______________   ________                          ____________    ____________          n
___ ______________   ________                          ____________    ____________          n
___ ______________   ________                          ____________    ____________          n
___ ______________   ________                          ____________    ____________          n
___ ______________   ________                          ____________    ____________          n
___ ______________   ________                          ____________    ____________          n
___ ______________   ________                          ____________    ____________          n
___ ______________   ________                          ____________    ____________          n
___ ______________   ________                          ____________    ____________          n
___ ______________   ________                          ____________    ____________          n
___ ______________   ________                          ____________    ____________          n
```

The following Vector Directory Number example shows the VDN form for VDN 51110, the VDN that is used in this BSR Application Plan example. In the example, the entry in the `BSR Application` field links this VDN to BSR Application Plan 10. Also note the `EAD-MIA` entry in the `BSR Available Agent Strategy` field. If vector 100 uses BSR commands, calls are not automatically delivered to the first resource found with an available agent. In each consider sequence, when the **`queue-to best`** or **`check best`** step executes, one of the following things happens:

If …	Then…
No skill has an available agent	The call is queued to the skill with the lowest adjusted EWT.
Only one skill has an available agent	The call is delivered to that skill.

If …	Then…
Two or more skills have available agents	The call is delivered to the skill with the most expert agent, which is the agent with the lowest skill level.
Two or more skills have available agents with the same skill level	The call is delivered to the skills that has the most idle agent.

Also note that `Allow VDN Override?` is set to n. If a second VDN and vector are used to process this call, the, the EAD-MIA strategy and the application plan that is specified for VDN 51110 is still used.

BSR example of primary VDN

```
change vdn xxxxx                                                    page 1 of 2
                              VECTOR DIRECTORY NUMBER

                                   Extension: 51110
                                        Name: Multi-site BSR
                               Vector Number: 100
                        Attendant Vectoring? n
                         Meet-me Conference? n
                         Allow VDN Override? n
                                         COR: 59
                                          TN: 1
                                    Measured: none
       Acceptable Service Level (sec): 20
        VDN of Origin Annc. Extension: 1001
                                   1st Skill:
                                   2nd Skill:
                                   3rd Skill:
```

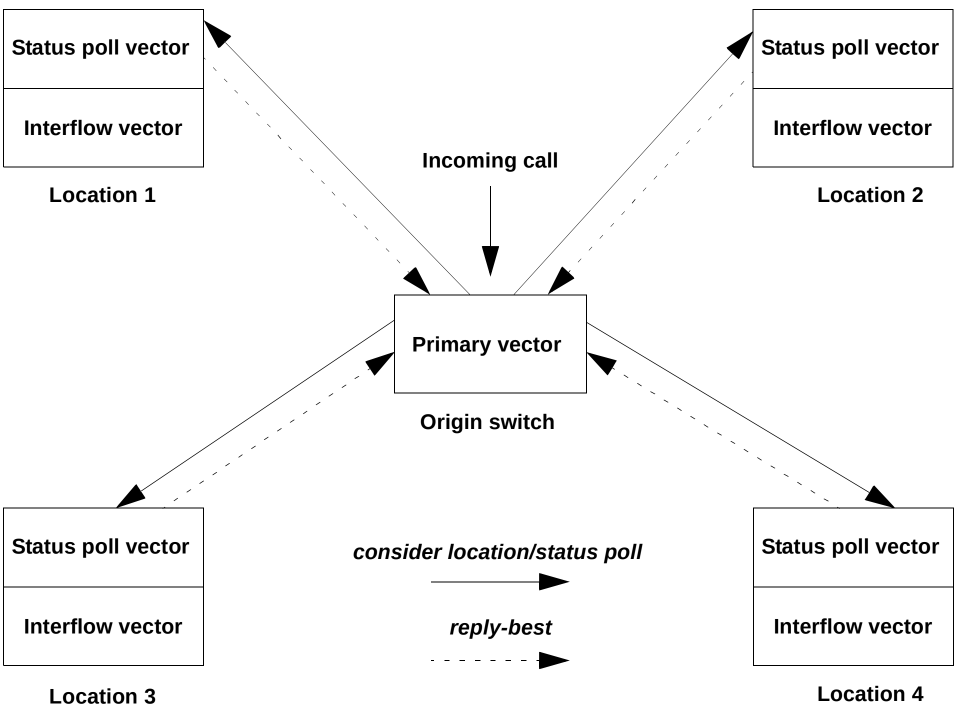
```
change vdn xxxxx                                                   page 2 of 2
                              VECTOR DIRECTORY NUMBER

                                  Audix Name:
                       Messaging Server Name:
                          Return Destination:
                      VDN Timed ACW Interval:
                             BSR Application:15
          BSR Available Agent Strategy: UCD-MIA
                    Observe on Agent Answer?:n
                      Conference Access Code:
                       Conference Controller:
          Display VDN for Route-To DAC?
```

With four remote switches to be considered, the overall application is represented in the following figure. Application plan 10 on the origin switch identifies the remote switches and provides the digit strings to dial into the VDNs for both the status poll vector and the interflow vector on each switch.

Each **`consider location`** command in the primary vector places a status poll call to its specified location. The status poll vector at that location executes a series of **`consider skill`** commands and returns data on the best resource to the origin switch through a **`reply-best`** command.

BSR example of multi-site application with four switches and limited tie trunk capacity



The following example shows the primary vector for this application. The first consider series in the primary vector tests two local skills. If either skill has an available agent, step 4 jumps to step 9 and the call is queued locally. No remote locations are polled. If no agents are available in either local skill, though, steps 5 to 8 test 4 remote locations. In general, you should not put other commands between **`consider`** steps. This use of the **`goto`** step is one of the few exceptions to that rule.

If the best remote location's adjusted EWT can reduce the call's current adjusted EWT, step 9 interflows the call to that location. In this vector, a local available agent is always favored over a remote available agent. Whichever location services a call, it will always be directed to the most idle, best skilled agent available.

Multi-site BSR example

```
1.   wait time 0 secs hearing ringback
2.   consider skill 1 pri m             adjust-by  0
3.   consider skill  2 pri m            adjust-by 20
4.   goto step 9 if expected-wait for skill best = 0
5.   consider location 1                adjust-by 30
6.   consider location 2                adjust-by 30
7.   consider location 3                adjust-by 50
8.   consider location 4                adjust-by 50
9.   queue-to best
10.  announcement 1001
11.  wait time 60 secs hearing music
12.  goto step 10 if unconditionally
```

In the primary vector, note that user adjustments are entered for local skill 2 as well as for all the remote locations. These indicate the administrator's preferences regarding both local and remote resources. For this example, let's say that neither local resource has an available agent and therefore an EWT greater than 0.

Status poll vector

Each receiving switch in a multi-site application must have a status poll vector. To collect information from these locations, each **`consider location`** command in the primary vector places a status poll to the status poll vector for the appropriate switch. The following example shows the status poll vector on the switch at location 3.

BSR example of status poll vector at location 3

```
1.   consider skill 2          pri m     adjust-by 0
2.   consider skill 11         pri m     adjust-by 20
3.   consider skill 21         pri m     adjust-by 30
4.   reply-best
```

This vector compares skills 2, 11, and 21, identifies the best one, and sends this information back to the origin switch through the **`reply-best`** command. Notice that user adjustments are applied to skills 11 and 21 to adjust the skill's EWT. When EWT adjustments are applied at both the origin and remote switches, the two adjustments are added at the origin switch. See User adjustments in multi-site BSR for more detail on user adjustments in multi-site applications.

In this example, suppose that skill 11 has the best adjusted EWT at location 3. Its data, including a user adjustment of 20, is returned to the origin switch by the **`reply-best`** command.

Finding the best resource

Once the remote switches have returned the best data for each location, the second consider series in the primary vector can be completed. In this example, let's suppose that no agents are available at any remote location.

The following table shows how user adjustments at the origin and remote switches yield the adjusted EWT for each location.

BSR best resource user adjustments

Location	Actual EWT of remote best (sec.)	User adjustment on origin switch	User adjustment on remote switch	Adjustment applied by origin switch (sec.)	Adjusted EWT used in BSR calculations (sec.)
1	60	30	0	30	90
2	45	30	10	40	85
3	40	50	20	70	110
4	70	50	0	50	120

The second consider series identifies location 2 as the best remote location, with an adjusted EWT of 85, and the **`queue-to best`** step interflows this call to location 2.

Interflow vector

The interflow vector on a remote switch in a multi-site application accepts the interflowed call from the origin switch. It also executes the same consider series as the status poll vector to identify the current best resource, in case conditions have changed since the status poll.

The following example shows the interflow vector on a remote switch.

BSR example of interflow vector at location 2

```
1.   consider skill   2       pri m                 adjust-by 0
2.   consider skill  11       pri m                 adjust-by 20
3.   consider skill  21       pri m                 adjust-by 30
4.   queue-to best
```

As happens today when a call is interflowed, it is removed from any queues at the origin switch and any audible feedback at the origin switch is terminated.

CAUTION:

BSR will not operate correctly unless the consider series in the status poll vector and the interflow vector use the same splits/skills with the same queue priorities.

Example of multi-site BSR with slow networks

Network response times are not an issue for most users. This example is intended for those users, if any, who experience such a problem. This example uses the same VDN, application plan, and four-switch network that is described in the Example of multi-site BSR with limited trunking on page 266. The vector in that example minimized interflows by using a **goto** step that skips the remote consider series if a local resource has an available agent. This design is especially useful if network response times are slow. Calls are always queued once locally before remote locations are considered.

Furthermore, both status polls and interflows are conditional. The call can wait in the queue for a local resource while BSR looks for a better split/skill at remote locations.

This example also shows the function of the **check best** command and the **wait-improved** conditional.

The following example shows the primary vector for this application, vector 100. The first consider series in the primary vector tests two local splits and queues the call to the best one. If the EWT for the best split is 30 seconds or less, step 5 jumps to the loop in step 11 and the second consider series is not executed. If the EWT for the best split is over 30 seconds, though, steps 6 through 9 test 4 remote locations. If the best remote location can reduce the call's EWT by more than 30 seconds as compared to its EWT in the best local queue, step 10 interflows the call to that location.

CAUTION:

Be certain to queue calls at least once before using the wait-improved conditional in a vector step. If calls are not already queued when the step with the wait-improved conditional executes, The switch reads the call's EWT as infinite. This could result in a vector that interflows all calls, even if that is not its intended function.

Multi-site BSR with EWT

```
1.   wait time 0 secs hearing ringback
2.   consider skill 1 pri m             adjust-by  0
3.   consider skill 2 pri m             adjust-by 20
4.   queue-to-best
5.   goto step 11 if expected-wait for call <= 30
6.   consider location 1                adjust-by 30
7.   consider location 2                adjust-by 30
8.   consider location 3                adjust-by 50
9.   consider location 4                adjust-by 50
10.  check best if wait-improved > 30
11.  announcement 1001
12.  wait time 60 secs hearing music
13.  goto step 11 if unconditionally
```

A consider series can end with either a **`queue-to best`** or a **`check best`** step. All consider series in the other examples have used a **`queue-to best`** command to queue the call unconditionally. The **`check best`** command lets you set conditions that must be met before a call is queued to the best resource. In this example, step 10 in the primary vector is **`check best if wait-improved > 30.`** In other words, step 10 interflows the call to the best location found by the consider series only if the EWT for that location is more than 30 seconds better than the call's EWT in the local queue.

You can use up to 3 consider series in one vector. It is possible to write more than 3 consider series in a vector, but there's no benefit in doing so. The switch only allows you to queue a call simultaneously to 3 different local resources. Since each consider series ends by queuing a call (assuming no agent is available), using more than 3 series in a vector will not place the calls in additional local queues. If the call interflows to another switch, it's removed from vector processing and any queues it was in on the origin switch.

It is also possible to combine single-site and multi-site consider series, as this example shows. Note that user adjustments are entered for local skill 2 as well as for locations 3 and 4. These indicate the administrator's preferences regarding both local and remote resources. In this example, say that step 2 queues the call to skill 1, which has an EWT of 65 seconds, before the second consider series is executed.

Status poll vector

Each receiving switch in a multi-site application must have a status poll vector. To collect information from these locations, each **`consider location`** command in the primary vector places a status poll to the status poll vector for the appropriate switch. The following example shows the status poll vector on the switch at location 3.

BSR example of status poll vector at location 3

```
1.   consider skill  2        pri m     adjust-by 0
2.   consider skill 11        pri m     adjust-by 20
3.   consider skill 21        pri m     adjust-by 30
4.   reply-best
```

This vector compares skills 2, 11, and 21, identifies the best one, and sends this information back to the origin switch through the **reply-best** command. Notice that user adjustments are applied to skills 11 and 21 to adjust the skill's EWT. When EWT adjustments are applied at both the origin and remote switches, the two adjustments are added at the origin switch. See User adjustments in multi-site BSR on page 265 for more details on user adjustments in multi-site applications.

Suppose that skill 11 has the best adjusted EWT at location 3. Its data, including a user adjustment of 20, is returned to the origin switch by the **reply-best** command.

Remember that the first consider series queued the call to local skill 1. Say that the second consider series identifies location 2 as the best remote resource. The **check** command in step 10 recalculates the call's current, unadjusted EWT in skill 1 and compares it to location 2's unadjusted EWT. If the call's actual (unadjusted) EWT can be improved by more than 30 seconds, the call is interflowed.

Note:

BSR uses adjusted EWT to determine which of the resources in a consider series is the best. Once the best resource is identified, subsequent **expected-wait** and **wait-improved** conditionals use the actual EWT values.

Interflow vector

When a call is interflowed to any of the remote locations, the interflow vector on that switch accepts the interflowed call from the origin switch. It also executes the same consider series as the status poll vector to identify the current best resource, in case conditions have changed since the status poll. The following example shows such an interflow vector.

BSR example of interflow vector at location 2

```
1.   consider skill  2         pri m     adjust-by 0
2.   consider skill 11         pri m     adjust-by 20
3.   consider skill 21         pri m     adjust-by 30
4.   reply-best
```

CAUTION:

BSR will not operate correctly unless the consider series in the status poll vector and the interflow vector use the same splits/skills with the same queue priorities.

Questions

If the call is queued to a remote resource by step 10 in the primary vector, is the call removed from the local queue that it entered in step 4?

When a call is interflowed, the call is removed from any queues at the origin switch and any audible feedback at the origin switch is terminated.

The second consider series can compare local and remote resources. If it does, and if step 10 queues the call to another local skill, will the call be removed from the local queue that it entered in step 4?

No. In general, the switch can queue a call to as many as 3 local splits/skills simultaneously. BSR does not change this limit.

Example for handling excessive wait times

This short example shows a simple primary vector in a multi-site BSR application. If wait times are sometimes excessive because of high call volumes, step 4 of this vector directs calls to a **`disconnect after announcement`** step when wait time in the network exceeds 5 minutes. The following example shows a simple primary vector.

Multi-site BSR using disconnect for excessive wait times

```
1.   wait 0
2.   consider skill 1         pri m               adjust-by 0
3.   consider location 2      pri m               adjust-by 30
4.   goto step 6 if expected-wait for best ≤ 300
5.   disconnect after announcement 3001
6.   queue-to best
```

Announcement 3001 might say something like, “We’re sorry. We are currently experiencing heavy call volume and cannot service your call at this time. Please try again later. We are normally least busy between 8 a.m. and 11 a.m. each morning.”

Planning and administering multi-Site BSR

This section presents information that is specific to BSR. Follow existing procedures to add or change other properties of VDNs and vectors not discussed in this section.

To create multi-site applications, follow the process below. List location numbers, Status Poll VDNs, and similar information so they will be available for planning and administration purposes. Define the purpose of the application

To define the purpose of the application:

1. Select the group of callers for which you want to create the application.
2. Define the goal of the application, for example, faster average speed of answer, better service by routing calls to the most qualified agents.
3. Decide which agent selection strategy (on VDNs) will best achieve your goal.
4. Decide whether you will implement BSR in a distributed system or a centralized system.
   - In a distributed system, all switches receive incoming calls and query other switches to interflow calls when appropriate.
   - In a centralized system, one switch serves as a hub. All incoming calls arrive at this switch and are routed from it to the other switches in the network.

   Since a distributed system is the more complicated of the two, the rest of this procedure is written in terms of implementing a distributed system. The same steps apply to implementing a centralized system, but only one switch will have application plans and primary VDN/vector pairs.

Select or create the elements of the application plan

To select or create the elements of the application plan:

1. Select the VDNs on each switch that serve the group of callers you have identified.

   On each switch these are the Primary VDNs for your application. You may, of course, want or need to create new VDNs. In either case, record the extensions of each VDN that will point to a vector with a BSR application.
2. Select the locations that you want to include in each application plan. To uniquely identify each location, assign a number between 1 and 255 and a short name of 15 characters or less.
3. Record the node number of the switch at each location.

4. Create Status Poll VDNs on each of the switches in the application plan.

   Record the full numbers you will need to route calls to these VDNs. These numbers will be entered on the Best Service Routing Application Plan form when you create the plan.

   If you are creating new VDNs on the switches that will receive interflowed calls, record these numbers too. You will need them to complete the BSR Application Plan form. Remember: you cannot use the same number for a Status Poll VDN and an Interflow VDN.

Administer the application on the switch

Define the application plan

To create an application plan on each switch:

1. At the command line prompt, type **`add best-service-routing`** ***`xxx`*** and press Enter (where ***xxx*** is a number between 1 and 255 that you want to assign to this BSR application.)

   The system displays the Best Service Routing Application Plan form. The number that you typed in the command appears in the `Application Number` field.

2. Assign a name to the plan.

   The best names are short and descriptive. This name cannot be longer than 15 characters.

3. Type in the information for the first remote location. Fill in the information for each field as shown below.

   Note:

   Each row on the form contains all of the information the BSR application needs to identify and communicate with one of the resources in the plan.

Fields on application plan form

Field	Type	Description
Num	Required	Type the number that you assigned to this location in 2.
Location Name	Optional	Type the name that you assigned to this location in 2.

Fields on application plan form (continued)

Field	Type	Description
Switch Node	Optional	This field is for user reference only. Leave it blank. If you are using the Universal Call ID feature, you may want to type each switch's node identity in this field. The switch's node identity is the number that is entered in the UCID Network Node ID field on page 4 of the Feature-Related System Parameters form.
Status Poll VDN	Required	This is the complete digit string that your switch will dial for the status poll call. The string can be up to 16 digits long.
Interflow VDN	Required	This is the complete digit string that your switch will dial to interflow a call to this location. The string can be up to 16 digits long.

4. Repeat 3 for each of the locations that you want to include in the application plan.
5. Press Enter to save your changes.

Note:

You must set up trunk groups to other sites. See Look-Ahead Interflow (LAI) on page 203 and Information Forwarding on page 151 for information on setting up trunk groups.

Link the application plan to a primary VDN and enter an agent selection strategy

To link the application plan to a primary VDN and enter an agent selection strategy:

1. Go to the Vector Directory Number form for the first VDN that you identified in 1.

   If this is a new application, create the VDN.

2. In the `Allow VDN Override?` field, type **y** or **n**. If the call is directed to another VDN during vector processing:

   — **y** allows the settings on the subsequent VDN, including its `BSR Available Agent Strategy`, to replace the settings on this VDN.

   — **n** allows the settings on this VDN, including its `BSR Available Agent Strategy`, to replace, or override, the settings on the subsequent VDN.

3. In the `BSR Application` field, type the application number you assigned to the plan.

4. In the `BSR Available Agent Strategy` field, type the identifier for the agent selection method you want this application to use:

If you enter…	The application will select the resource with…
1st-found	The lowest Expected Wait Time. If the application finds an available agent before it has compared all the locations in the plan, the application routes the call to that agent without contacting any other locations.
ucd-mia	The agent who has been idle the longest. The application compares all the locations in the plan.
ead-mia	The agent with the highest skill level, which is the lowest skill number, who has been idle the longest.
ucd-loa	The least-occupied agent.
ead-loa	The agent with the highest skill level, which is the lowest skill number, who is the least occupied.

5. Press **Enter** to save your changes.

Repeat 1 through 5 on each switch that needs an application plan and a Primary VDN/vector pair.

This process covers the administration that is needed for BSR vector commands to function. Now, of course, you need to write or modify the vectors that will control call processing.

Troubleshooting for multi-site BSR

You should regularly execute a **display events** command for the appropriate vectors, especially if you have just implemented a new BSR application. Vector events will identify and indicate the source of common malfunctions and administration errors.

When tie-trunks or queue slots become exhausted, BSR cannot effectively balance calls across the network. If such problems are revealed frequently by vector events, review the design of the BSR application involved. If tie-trunks are frequently exhausted, the user adjustments on **consider location** steps may be set too low.

For a list of BSR vector events and definitions, see Tracking unexpected events on page 512.

Note:

Only the most recent events are displayed when a **display events** command is executed. For this reason, you should periodically display vector events to help quickly identify problems.

To verify that your BSR vectors are operating as intended, use a **list trace vdn** or **list trace vec** command to observe processing of an individual call. See Clearing events on page 529 for more information.

BSR status poll vectors must always end with a **reply-best** step. A **busy** or **disconnect** command should never be used.

Tips for writing BSR vectors

Before you write your first vector using BSR, you should study the sample vectors that are provided and familiarize yourself with the new commands and command elements. Sample vectors are provided in Single-site BSR on page 237 and Multi-site BSR on page 253. The new commands and command elements are explained in Appendix A: Call Vectoring commands on page 387.

As you write BSR vectors, it is strongly recommended that you follow the guidelines below.

- Arrange your **`consider`** steps in order of preference.

  The **`consider`** step that tests the main, or preferred, resource should be the first in the series. The second **`consider`** step should test the resource that is your second preference for handling the given call type, and so on. To avoid unnecessary interflows, put **`consider`** steps for local resources before steps that consider remote resources. This arrangement also provides a local “best” as a backup in case the interflow fails.

  Arranging `consider` steps in order of preference is recommended for all BSR vectors. It is especially important when the active VDN for the call is using the 1st-found agent strategy since the switch delivers the call to the first available agent found, arranging **`consider`** steps in order of preference ensures that calls are delivered to the best of the available resources and that unnecessary interflows are avoided.

- Do not put any commands between the steps of a consider series that would cause a delay. **`Goto`** commands are OK.

- Do not put a consider series in vector loops.

- Confirm that calls queue successfully.

  This check is recommended for all vectors. Since EWT is infinite for a call that has not queued, a step that checks EWT after a queue attempt is a good confirmation method. After a **`queue-to best`** step, for example, a command such as **`goto step X if expected-wait for call < 9999`** should be included.

- Do not use the wait-improved conditional in a vector before you have queued the call at least once.

  The **`wait-improved`** conditional compares the call’s EWT in its current queue to the best resource that is found by a consider series. If a call has not been queued and a vector step such as **`check best if wait-improved > 30`** is executed, the switch interprets the call’s current EWT as infinite and the **`check best`** step always routes the call to the best resource. In other words, in this situation the **`check best`** step functions like an unconditional **`goto`** or **`route-to`** command.

BSR-initiated path-replacement for calls in vector processing

Path replacement for calls in queue and vector processing can be accomplished using QSIG or DCS with Reroute using ISDN SSE. For calls that are waiting in queue or in vector processing, even if the call is not connected to an answering user, path replacement can be attempted to find a more optimal path for this call. This results in more efficient use of the trunk facilities.

The **`queue-to best`** command is used in BSR to initiate a QSIG path replacement for a call. The following scenario can take place:

- At the terminating switch, if a Path Replacement Propose operation is received for a call that is in queue or vector processing, the switch can immediately initiate path replacement using the Path Replacement Extension if the `Path Replace While in Queue/Vectoring` field is set to `y` and the `Path Replacement Extension` field has a valid entry. These fields are located on the ISDN parameters page of the Feature-Related System Parameters form.

  Note:

  The Call Management System (CMS) load r3v9ag and earlier cannot track a measured ACD call after path replacement has taken place. CMS load r3v9ah and newer does keep the CMS call record of the measured ACD call intact after path replacement takes place.

Example vector

The following example shows how a BSR vector can be written to trigger path-replacement at the terminating switch.

Note:

In order for a path-replacement to be attempted, the incoming and outgoing trunks that are used for the call must be administered with the `Supplementary Service Protocol` field set to `b`.

BSR-initiated path-replacement vector

```
1.   wait 0
2.   consider skill 1
3.   consider skill 5
4.   consider location 10              adjust-by 10
5.   consider location 24              adjust-by 20
6.   queue-to best
```

At the terminating (receiving) switch, the vector that is executed by the incoming call must be programmed with an `announcement`, or **`wait hearing music`** vector command. The use of one of these commands is what makes it possible for path-replacement to take place while the call is in vector processing.

Chapter 13: Network Call Redirection

Contact centers are looking for many ways to reduce costs. One of these ways is to employ Virtual Private Networks (VPNs) to eliminate as much private network cost as possible. These cost reductions are particularly valuable in enterprises or multi-site call-center environments and especially to Enterprise contact centers where network costs are typically high. Network Call Redirection (NCR) offers an optional call redirection method between sites on a public network or a Virtual Private Network and to reduce trunking costs.

This chapter includes information on the following topics:

What is Network Call Redirection?

Call redirection using NCR is accomplished by using either the public network's NCD or NCT options. NCD "clear call upon invocation" is only offered outside of the United States. In the United States, only NCT is offered. In the future, the NCD "retain call until alerting/connect" option may be provided by public networks outside of the United States, but it is not currently available. NCR supports Information Forwarding via UUI transport to the redirected-to location.

Network Call Transfer

NCT occurs after the incoming call is initially answered. With NCT, the switch is required to set up the second leg of the call and then wait for the second site to acknowledge before requesting the PSTN to transfer the first leg of the call to the second leg, and before the PSTN drops the trunks to the Avaya switch. The benefit is that the switch retains control over the call and can redirect the call using the trunk-to-trunk method should the NCT invocation fail. Therefore, the NCT option is the most reliable.

After the second leg of the call is initiated and acknowledged by the public switch, the public network joins the original ISDN caller to the redirected-to endpoint and then drops both the original ISDN call and the second leg of the call at the redirecting switch.

Network Call Deflection

NCD occurs before the incoming call is initially answered. With NCD, the public network sets up the second leg of the call to the redirected-to location when the Avaya switch deflects the call. There are two PSTN options for NCD, per the ETSI standards: "retain call until alerting/connect" and "clear call upon invocation." This is commonly referred to as a partial call reroute.

With the "clear call on invocation," the switch loses control of the call once the call has been transferred to the public network for redirection. The switch does not retain control of the call until it has been acknowledged by the network, so there is no alternative transfer possible if the public switch cannot transfer the call to the second location.

The "retain call until alerting/connect" option is not widely available (no known PTSN offers it at this time). With this option, the PSTN sets up the second leg of the call and waits until an alerting message is received before the first leg of the call is dropped. In this case, if the second leg of the call fails, then the switch can redirect the call through another method (such as trunk-to-trunk connection) and not lose the call.

Limitations on call redirection

There may be limits placed on the number of times a call may be redirected over the public network. These limits are imposed by the public network service provider. For example, in the United States, MCI currently allows only one redirection per call. In the United Kingdom, there is a limit of 20 call deflections per call. In addition, there may be additional charges associated with redirected calls.

Additionally, some public network service providers do not support forwarding of User-to-User Information (UUI), including ASAI User data, collected digits, VDN name, the VDN in-time (as reflected by the NETINTIME database items), and the UCID. This means that Information Forwarding will be lost and the second leg of the redirected call will look like an entirely new call to the redirected-to switch at the second location. One of the data items lost is the VDN name, which is rerouted to the originally called service (DNIS) information. The indication that the call has been forwarded can be achieved by using dedicated VDNs for call forwarding, but it does reduce the benefits of Information Forwarding inherent with NCR. Also, this option limits CTI applications as there is no ASAI information or UCID forwarded.

At this time, no PSTNs are offering the Network Call Deflection “retain call until alerting/connect” operation. Therefore, only the Network Call Deflection “clear call upon invocation” offer is available from PSTNs. Both methods are described in this document. It is advised that you negotiate with your PSTN as the NCR feature will work on either platform. NCR is limited by which PSTN platform is available to you.

Information Forwarding support for AT&T In-band Transfer and Connect

Enabling NCR also provides Information Forwarding support for the AT&T Transfer and Connect In-band network service ISDN D-channel data forwarding capability. The Information Forwarding feature forwards UUI that is associated with the call to the “transferred to” location. When NCR is active in the switch system, transferring the call using Call Vectoring and AT&T In-band Transfer and Connect, the **`disconnect`** vector step includes the codeset UUI IE in the ISDN DISSCONNECT message.

NCR considerations

This section includes important information that you need to know prior to beginning use of NCR. The issues addressed include:

- Compliance
- Trunking considerations
- Station Call Transfer/Conference
- Information Forwarding

Compliance

The Network Call Deflection feature is compliant with ETSI Supplementary Services Network Call Deflection ETS 300 207-1 (partial call rerouting in the public network).

The Network Call Transfer feature is compliant with ANSI Explicit Network Call Transfer (ENCT) T1.643 (1995), the MCI Nortel DMS-250 variant of ANSI ECT (1995), Telcordia Two B-Channel Transfer (TBCT) and the 1998 version of ANSI ECT.

Support for NCR on PSTNs varies with geographical location, and support may be limited or absent in some areas. Consult your Avaya account team to determine availability in your area.

Note:

The compliance requirements are needed to negotiate service with your PSTN. There are PSTN-imposed limits from country to country as to how many times a call can be redirected. Also, PSTNs typically charge for the redirect and UUI transport services.

Trunking considerations

The MCI network requires that the calls being transferred are on the same Direct Access Line (DAL). The Avaya switch enforces this requirement by only requesting NCT for two calls using the same signaling group. This will result in sending the request on the same D-channel used for the first call with associated signaling or on the associated D-channel when active with Non-Facility Associated Signaling (NFAS) D-channel backup configuration. This requirement is also imposed for the ANSI and TBCT forms of NCT.

With vector-invoked NCT (BSR or **`route-to-number`**), the second leg of the call is placed over an idle trunk in the same trunk group as the incoming call to ensure that invocation of NCT is done over the same signaling group. Therefore, vectoring activation of NCT requires that the trunk group be a two-way trunk group.

Call-by-Call Service and Usage Allocation can be used to reserve a number of trunks in the two-way trunk group. Use the trunk group Call-by-Call Service Selection (CBC) Usage Allocation capabilities to reserve several trunks for outgoing calls. Set the incoming call NSF (Network Specific Service) types minimum channels to the amount to be reserved and the maximum to the total trunks, less the amount reserved.

With station, ACD agent, VRU, or CTI-initiated conference/transfer, if the second leg of the call is set up over an outgoing trunk with the same signaling group as the incoming call, then NCT can be invoked when the transfer is requested. If an outgoing trunk group is assigned to the same signaling group as the incoming trunk group and that outgoing trunk is selected by the outgoing call (using ARS) placed by the user/CTI application for transfer, then a two-way trunk group is not required.

Note:

With incoming 800 number calls from MCI WorldCom DMS-250 network switches, the second leg call must be answered (ISDN connect message received) before NCT can be invoked.

Implementing and administering NCR

The NCR feature uses either the Network Call Transfer (NCT) or Network Call Deflection (NCD) operations provided by the PSTN to redirect an incoming ISDN call from the Avaya switch to another PSTN endpoint. In the call center environment, NCR is intended for multi-site configurations where ISDN calls are interflowed between switches over the PSTN by the Best Service Routing (BSR) feature's **`queue-to-best`** vector step, which provides the best approach for balancing loads across a multi-site environment and is more cost effective and accurate than pre-delivery routers. See Best Service Routing (BSR) on page 229 for complete information on BSR.

The NCR feature can also be used to redirect an incoming ISDN call by either of the following methods:

- As a substitute for the interflow over trunk-to-trunk tandem connections or other non-attendant call vectoring applications using the `~r` **`route-to-number`** vector step
- Attendant call vectoring, by using the `~r` **`route-to-number`** vector step
- ASAI Third-Party Merge/Call Transfer Operation (Network Call Transfer only)
- Station transfer by DCP set Transfer button/hangup or analog station switch hook flash transfer by hangup
- Station transfer by DCP set Conference button, in which the conferencing (middle) party connects the two calls and then hangs up

The NCR feature is designed to optimize the rerouting of ISDN calls over the public network since no switch trunks are retained at the redirecting switch after the call is rerouted.

NCR may be activated and tracked with ASAI/CTI. The ASAI event reporting capabilities allow tracking of the NCR-redirected calls by their Universal Call ID or ASAI User-to-User Information.

Network Call Redirection is an optional switch feature.

NCR may only be activated for incoming ISDN trunk calls where the associated trunk group has been enabled by the public network service provider to use Network Call Transfer or Network Call Deflection features. Also, NCR supports Information Forwarding for AT&T In-band Transfer and Connect network service.

Note:

Network Call Transfer (NCT) currently works with only the MCI WorldCom DMS-250 network switches. NCT is not currently offered on MCI DEX600 switches. Until NCR has been tested on specific PSTNs, performance is not guaranteed. To verify operability, contact your CRM Regional Offer Manager.

NCR activation using Call Vectoring

If NCR is activated using either the **`route-to-number`** or **`queue-to-best`** vector steps, either the NCT or the NCD options may be used to redirect an incoming call while the call is still being processed by the call vector.

The NCR feature is activated by call vectoring on the switch if:

- The Best Service Routing feature selection of a best location has been administered with the `Net Redir?` option set to `y` on the BSR Application Table form (with both BSR and LAI active), followed by the execution of the **`queue-to-best`** vector step. See *Avaya MultiVantage Call Center Software – Guide to ACD Call Center*, 555-230-716, for more information.
- The **`route-to-number`** vector step is administered with a `~r` as the first item in the number field (with or without the `LAI` option set to `y` or with Attendant Call Vectoring active).

Network Call Transfer Activation

If the NCT feature is enabled for the trunk over which an incoming call is routed to the switch, then NCR redirection will be attempted only if a CONNect ISDN message (answer supervision) has been sent to the public network for the original call. Any vector step of this type, such as **`wait hearing music`** or **`collect x digits`** (refer to the list below) will send the CONNect ISDN message to the public network and NCR will be attempted with either the **`route-to-number`** or **`queue-to-best`** vector steps. If none of the vector steps listed below have been executed, then the switch will automatically send a CONNect ISDN message to the public network before call transfer is requested from the PSTN. This will start PSTN charging for the call, which is done after the second leg call is established.

If NCT is used, a second call is set up by the redirecting switch to redirect the call using the public network. That call must be answered (sends a CONNECT ISDN message) by the redirected-to location before the call will be requested from the PSTN. This is necessary for reliable NCT operation with MCI's network switches.

Network Call Transfer call success

With NCT, a call transfer is considered to be successful if:

- The public network responds to the FACility message that requested the NCT operation with a FACility message to the requesting switch indicating “PSTN success.” The requesting switch should then receive a DISConnect message for both the first and second leg of the call after the original call and the second leg are joined by the PSTN.
- If the public network responds with a “PSTN failure” FACility message to the requesting switch, which will result in a trunk-to-trunk connection between the first leg of the call and the second leg of the call. Vector call processing considers this to be successful for the NCR attempt because the original call was not lost. However, a vector event will be generated indicating that the NCT public network operation failed, and this call will not be indicated as “deflected” in CMS.

Network Call Deflection activation

NCD may only be activated if a CONNect ISDN message has not been sent to the public network for the call; this is call answer supervision. In that case, the following vector steps should not be used by a vector in prior steps if the NCD feature is going to be used for NCR:

- **`wait hearing music`**
- **`collect x digits`**
- **`announcement`**
- **`converse-on split`**

If NCD is used, the switch does not set up a second call to redirect the call on the public network, and only the incoming ISDN D-channel is used by the public network to redirect the call. The second call is actually set up by the public network.

Network Call Deflection call success

With NCD, verification of a successful NCR attempt varies with the rules set up for the incoming ISDN trunk:

- If a subscription to the CD “clear call on invocation public network” service is in effect for the incoming ISDN call’s trunk group, a successful NCR attempt is signified when the public network has validated the NCR request and returned a call reroute return result in a DISConnect message.
- If a subscription to the NCD “retain call until alerting/connect” operation is in effect for the incoming ISDN call’s trunk group, a successful NCR attempt is indicated when the public network sends a call reroute return result in a FACility message, followed by a DISConnect message to the requesting switch for the first leg of the call. In this case, the DISConnect message is sent only after the PSTN has received an ALERTing or CONNect ISDN message from the redirected-to public network endpoint to indicate that call deflection is successful.

NCR activation using BSR vector processing

Network Call Redirection of ISDN calls are performed if the following vector administration has occurred within Best Service Routing. Network Call Redirection on the Customer Options form must have been set to Y for this installation to be able to administer NCR:

1. An incoming ISDN call has entered vector processing.
2. The call may or may not encounter a vector step, such as an announcement, that causes an answer ISDN message (the ISDN CONNect message) to be returned to the trunk associated with the call.
3. The call has encountered one or more **`consider location BSR`** vector steps that returned valid data (such as Expected Wait Time) and then executed a **`queue-to-best BSR`** vector step.
4. BSR call processing has determined that the call should be interflowed to one of the remote BSR locations previously considered as the best location.
5. The interflow VDN number in the BSR Application Table associated with the best location has been administered with the `Net Redir?` field set to y.
6. The administered interflow number is the public network phone number (without trunk/ARS/AAR access codes) that reaches the remote call center site. The long distance access (dial 1 in the United States) may also be required depending on the PSTN requirements for the trunk group. This number is used differently during the invoke process depending on which redirection service is available from the public network service provider.
7. With call vectoring activation of NCR, either NCT or NCD will be initiated based on which type of redirection is administered to the trunk group for the incoming call. In either case, a FACility message is sent to the public network over the D-channel associated with the incoming trunk to invoke redirection of the call.

A successful NCR completion (when the PSTN indicates a successful NCT invoke) terminates vector call processing while the original call is connected through the interflowed to call center site by the public network. CMS will track this as a successful network call redirection for the call.

An unsuccessful NCR attempt (NCD invoke failure or NCT secondary call failure) results in vector processing going to the next step in the vector following the **`queue-to-best`** vector step prior to Release 10. Starting with load 37 of Release 10, the call will queue to a local best split/skill, if one has been considered, and it has staffed agents. NCT invocation failure after the second call is established results in the switch reverting to the trunk-to-trunk call connection.

Using BSR with Network Call Transfer

The following steps are used when the ~r **`route-to-number`** vector step is used for Network Call Redirection:

1. The call arrives at the first location.
2. The call is processed by a vector that has a `~r` in the leftmost two character positions in the number field or the `Network Redir?` field on BSR Application Table form is set to `y` for the location specified in the **`consider`** step for the active VDN application.
3. The switch sends the call to the public network.
4. The public network switch sets up the second leg of the call and passes the codeset 0 UUI information in the SETUP message if this is supported.
5. The switch tells the public switch to transfer the call over the public network.

   For incoming 800 number calls from MCI WorldCom DMS-250 network switches, the vector reached by the second leg call placed by Avaya switch must immediately be answered (and send an ISDN CONNect message). This can be accomplished via a **`wait 0 secs hearing music`** or an **`announcement`** step as the first step in the interflow receiving vector.
6. The public network merges the second leg of the call to the second site and drops the Avaya switch. With NCT, at this point, if the second call fails, the Avaya switch can maintain control of the call and revert to trunk-to-trunk transfer.

NCR activation by route-to number vector processing

The following steps are used when the ~r **`route-to-number`** vector step is used for NCR:

1. The call arrives at the first location.
2. The call is processed by a vector that has a ~r in the leftmost two character positions in the number field, or the `Network Redir?` field on the BSR Application Table form is set to `y` for the location that is specified in the consider step for the active VDN application.
3. The Avaya switch sets up the second leg of the call using the public network and passes the codeset 0 UUI information in the SETUP message if this is supported. This call is then answered at the remote end. With NCD, the second leg of the call is set up by the public network.
4. The Avaya switch tells the public switch to transfer the call over the public network.
5. The public network merges the second leg of the call to the second site and drops the switch. With NCT, at this point, if the second call fails, the Avaya switch can maintain control of the call and revert to trunk-to-trunk transfer.

Sample vectors

BSR vector with NCR

```
wait 2 seconds hearing ringback
consider skill 1 pri l adjust-by 0
consider location 1 adjust-by 20
consider location 2 adjust-by 40
queue-to-best
```

If the `Network Redir?` field is set to `y` for the best BSR location, then the NCR feature is activated.

The following vectors are examples of vectors administered using the **`route-to-number`** command to use NCR.

Sample ACD vector

```
wait 0 seconds hearing ringback
goto step 4 if skill oldest-call < 30 secs
route-to number ~r13035403001
queue-to skill 35 priority m
...
```

Sample Attendant vector

```
goto step 6 if time-of-day is all 17:00 to 09:00
wait 0 seconds hearing ringback
queue-to attd-group
wait 999 secs hearing music
stop
route-to number ~r13035551002
```

NCR and ASAI

NCR is activated by ASAI call processing when the Third-Party Merge/Call Transfer operation is requested by a CTI operation. This occurs in the following manner:

1. This is typically initiated by the CTI user selecting an icon, menu item, or button to transfer an incoming ISDN call to another user on the public network.

   Since the incoming ISDN call must be connected to a station user before the Third-Party Merge/Call Transfer operation is requested, NCR can only initiate the call redirection if NCT is optioned on the trunk.

   With incoming MCI WorldCom DMS-250 network switch 800 calls, the far end call must be answered first.

2. If a call arrives at an ASAI monitored VDN and either the NCT or NCD feature is used, then ASAI will send appropriate information in the disconnect event to tell the application that the call has been redirected by NCR.

ASAI event reporting allows tracking of ISDN ACD calls that were redirected by NCR in a multi-switch ECS call center environment. These calls can be tracked by the UCID assigned to each call, or by the UUI information inserted by the application through either the Third Party Make Call or Adjunct Routing features.

Station call transfer/conference

An incoming ISDN call (over a trunk with NCT PSTN service) is answered at the station or voice response unit (VRU or IVR). The station user/VRU answers the call and initiates a station call transfer using the Transfer feature button or a switch hook flash. The switch automatically sends the `invoke NCT` ISDN FACility message when the transfer is complete if:

- NCT is assigned to the incoming trunk group
- the call is eligible for NCT — that is, if the second leg of the call has been set up over a trunk with the same signaling group as the incoming call.

Note:

If the station user initiates and completes a three-way conference instead, the switch automatically sends an `invoke NCT` ISDN message when the initiating station user drops from the three-way conference.

The following steps provide additional information about NCR activation using station call transfer or conference:

1. An incoming ISDN call (over trunk with NCT PSTN service) is answered at the switch station/VRU line port.
2. A station or ACD agent user initiates station call transfer using feature button or switch-hook flash
3. VRU (out of vector processing) initiates station call transfer using a switch-hook flash.
4. The switch automatically sends an `invoke NCT` ISDN FACility message when the transfer is completed after the second leg is set up.
5. If the station user initiates and completes a three-way conference instead, the switch automatically sends an `invoke NCT` ISDN message when the initiating station/VRU user drops from the three-way conference.

The initiator (station/agent user or VRU) dials the second leg connection by using an access code plus the PSTN number after initiating the transfer. The access code must select an idle outgoing trunk in a trunk group with the same signaling group as the incoming call with NCT active. The transfer is completed when the Transfer button is pressed or the initiator hangs up. With incoming MCI WorldCom DMS-250 network switch 800 calls, the far end call must be answered first.

CTI/Station Transfer considerations for administration

The NCR feature is activated automatically for a station-user or CTI call transfer under the following conditions:

- The `ISDN Network Call Redirection` field is set on the System Parameters Customer Options form.
- The second leg of the call transfer for an incoming ISDN call is made using the same trunk group with a trunk that has the same D-channel as the incoming call.

To allow the PSTN switch to complete the station-user or CTI invocation of NCT operation successfully, the PSTN number that a station-user of CTI would dial to transfer an incoming call to another PSTN endpoint must be added (in some fashion) to the ARS digit analysis form. For the routing pattern associated with the ARS digit analysis form entry, the following settings must also be administered in an entry line on the lower part of the route-pattern form:

- `Service/Feature` field = `sdn`
- `Number Format` = `lev110-pvt`

Also, the PSTN service provider should be contacted to verify that the configuration of the PSTN switch used for the Network Call Transfer operation has been properly configured to accept the outgoing digits used by the Avaya switch station-user or CTI application to set up the second leg of the call transfer.

NCR and Information Forwarding

The Information Forwarding feature is supported with NCR when the PSTN supports Type 1 MA-UUO transport in conjunction with the network redirection feature. MCI supports UUI transport via the N-Quest Type 1 service currently only available with MCI WorldCom DMS-250 network switches.

The Information Forwarding feature forwards call center-related data (as User-to-User Information) with the call including:

- ASAI user data
- Universal Call ID (UCID)
- Collected digits
- In-VDN time
- VDN name.

With NCD, the UUI is included in the FACility invoke message and the PSTN forwards the UUI to the second site.

With NCT, the UUI is included by the Avaya switch in the SETUP message to the second site.

With NCD, Information Forwarding information is forwarded in the ISDN FAC message to the PSTN.

With NCT, Information Forwarding information is forwarded in the ISDN SETUP message for the second leg call.

With either NCD or NCT, the PSTN facility must support MA-UUO (the N-Quest Type 1 service available with MCI WorldCom DMS-250 network switches).

NCR support for AT&T In-band Transfer and Connect

NCR supports Information Forwarding for the AT&T Network In-band (IB) Transfer and Connect service. The Transfer and Connect service is a method that the AT&T network uses to transfer a call within the network and drop the original trunk to the first destination by using in-band DTMF (touchtone) dialing.

The Transfer and Connect Courtesy Transfer IB trigger feature allows the redirecting party to forward user data with the transferred call using out of band data forwarding in the ISDN PRI DISCONNECT message via message-associated UUI signaling over the D-channel. The data is forwarded with the network transferred call if the Customer Premises Equipment (CPE) switch includes a codeset 0 or 7 UUI IE in an ISDN DISCONNECT message sent by the CPE within three seconds of the DTMF transfer request digits. See AT&T TR 50075 for details. With NCR, the Information Forwarding feature forwards either the ASAI user data (with the Service Provider setting) or the full call center data set (with the Shared setting) when a call is redirected to another location.

UUI forwarding

Activating NCR allows user data to be included in the ISDN DISCONNECT message when a vectoring **`disconnect`** step is processed. The user data is included in a codeset 0 UUI IE, which is then forwarded to the transferred-to remote location. The user data can then be handled by the remote switch in the same manner as an LAI/BSR interflowed call. The inclusion of the UUI IE only occurs when the disconnect step does not have an announcement specified.

This data forwarding capability can be used with application that invoke Courtesy Transfer using an **`announcement`** step in vectors followed by a **`disconnect`** step. The announcement has the in-band DTMF transfer trigger (*8) and the transfer to direct dial telephone number digits recorded.

Chapter 14: Attendant Vectoring

The Attendant Vectoring feature enables a set of commands that can be used to write call vectors for calls to be routed in non-call center environments. When Attendant Vectoring is enabled, all attendant-seeking or “dial 0” calls are processed using the call vectors, not the normal attendant console call routing.

The main reason to use Attendant Vectoring is to allow flexible routing of attendant-seeking calls. If users are instructed to dial an attendant VDN, the call could be answered by an attendant, but it may also be covered to the voice mailbox of a night station. Training users to understand these different call routing options is something you should consider before using Attendant Vectoring.

If you use Attendant Vectoring and night service to route calls to a voice mail system, you can also use the Automatic Message Waiting (AMW) feature to notify after-hours personnel that there are messages in the night service station mailbox by assigning an AMW lamp on one or more backup telephones. When personnel see that there are new messages, they can check those messages after hours and act upon them as needed.

This chapter includes the following sections:

Command set

The following table lists the commands associated with Attendant Vectoring.

Attendant vectoring command set

Command category	Action taken	Command
Treatment		
	Play an announcement.	**`announcement`**
	Play a busy tone and stop vector processing.	**`busy`**
	Disconnect the call.	**`disconnect`**
	Delay with audible feedback of silence, ringback, system music, or alternate audio/music source.	**`wait-time`**
Routing		
	Queue the call to an attendant group.	**`queue-to attd-group`**
	Queue the call to an attendant extension.	**`queue-to attendant`**
	Queue the call to a hunt group.	**`queue-to hunt-group`**
	Route the call to a specific extension number.	**`route-to number`**
Branching/programming		
	Go to a vector step.	**`goto step`**
	Go to another vector.	**`goto vector`**
	Stop vector processing.	**`stop`**

Treatment commands

Attendant Vectoring allows use of several TREATMENT commands, including:

- announcement command *on page 301*
- busy command *on page 301*
- disconnect command *on page 301*
- wait-time command on page 301

The following sections detail the syntax that can be used for these commands and any information that is specific to their use in Attendant Vectoring.

announcement command

Syntax: announcement <extension>

The usage for the **announcement** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

busy command

Syntax: busy

The usage for the **busy** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

disconnect command

Syntax: disconnect after announcement <extension>

The usage for the **disconnect** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

wait-time command

Syntax: wait-time <time> secs hearing <silence, ringback, music>

This use of the **wait-time** command was slightly modified for attendant vector usage. The **i-silent** treatment choice was removed because it does not pertain to attendant vectoring. The **wait-time <seconds> secs hearing <extension> then <silence, ringback, music, continue>** command was left unchanged. No other changes or attendant specific considerations apply, so these commands work as they do in Basic Call Vectoring.

Routing commands

Attendant Vectoring allows use of several ROUTING commands, including:

- queue-to attd-group command *on page 302*
- queue-to attendant command *on page 303*
- queue-to hunt-group command *on page 304*
- route-to number command on page 304

Note:

A **`wait-time 0 secs hearing ringback`** step should be used to give immediate feedback to the caller. The **`queue-to`** command does not provide ringback until the call is actually ringing the attendant. The **`wait-time`** step should be implemented as the first vector step or as the step immediately before the **`queue-to`** step.

The following sections detail the syntax that can be used for these commands and any information that is specific to their use in Attendant Vectoring.

queue-to attd-group command

Syntax: queue-to attd-group

The **`queue-to attd-group`** vectoring command is available only for attendant vectors. If an attendant group call is redirected to vector processing that queues the call to the attendant group, the group to which the call gets queued is determined by the TN assignment that is associated with the call. If an attendant in the group is available to take the call, it is terminated to the attendant, not queued, and vector processing terminates.

Attendant group based on tenant number

When attendant group calls are redirected to vector processing and are programmed to queue to the attendant group, the attendant group is the group that is designated for the call's associated tenant number.

If an attendant group call is redirected to vector processing that queues the call to the attendant group, the call is placed in the queue using the priority that is assigned for the call. Attendant queue priorities are assigned on a system-wide basis, not on an individual partition basis.

Attendant group queue

Calls that are queued to the attendant group by way of attendant vector processing are queued with the system-administered priority for the call. If an attempt is made to queue the call and it fails, the vector event for queue failure is logged.

As with other vector queue commands, vector processing continues with the next step following the **`queue-to attd-group`** command regardless of success or failure. The **`goto step if queue-fail`** command is provided for handling failure conditions. Otherwise, on success, announcements or other feedback can be applied while the call is in queue. Other than the provision of caller feedback, attendant queue functionality is unchanged. If no commands follow a successful queue step, the call is left in the queue with no feedback. If no commands follow a failed queue step, the call is dropped. Anytime the end of vector processing is reached without the call being placed in queue, it is dropped and an event is logged.

queue-to attendant command

Syntax: queue-to attendant <extension>

The **`queue-to attendant`** vectoring command is available only for attendant vectors. If an attendant group call is redirected to vector processing that queues the call to an individual attendant, the attendant to whom the call gets queued must be a member of the attendant group that is indicated by the TN assignment associated with the call. If the attendant is available to take the call, the call is terminated to the attendant, not queued, and vector processing terminates.

The success of this command depends on having individual attendant access. These calls are queued based on the priority that is assigned to individual attendant access calls.

Individual attendant queue

Calls that are queued to the individual attendant via attendant vector processing are queued with the system-administered priority for individual attendant access calls. If the indicated attendant is not a member of the associated attendant group, the command is considered failed and vector processing continues with the next vector step. If an attempt is made to queue the call and it fails, a vector event is logged.

As with other vector queue commands, vector processing continues with the next step following the **`queue-to attendant`** command regardless of success or failure. The **`goto step if queue-fail`** command is provided for handling failure conditions. Otherwise, on success, announcements or other feedback can be applied while the call is in the queue. If no commands follow a successful queue step, the call is left in the queue with no feedback. If no commands follow a failed queue step, the call is dropped. Anytime the end of vector processing is reached without the call being placed in queue, the call is dropped and an event is logged.

queue-to hunt-group command

Syntax: queue-to hunt-group <#> pri <l (low), m (medium), h (high), t (top)>

This vectoring command is available only for attendant vectors. However, it is the functional equivalent of the split queueing command. As such, a call can be queued to up to three hunt groups. If an attendant group call is redirected to vector processing that queues the call to a hunt group, the call is queued with the indicated priority. If a hunt group member is available to take the call, it is terminated to the member, not queued, and vector processing terminates. In order to use a hunt group in vectoring, it must be administered as a vector controlled group. However, it can be any type of hunt group, including UCD, ACD, and so forth.

Hunt group queue

Calls that are queued to a hunt group by way of attendant vector processing are queued with the indicated priority for the call. If an attempt is made to queue the call and it fails, a vector event is logged.

As with other vector queue commands, vector processing continues with the next step following the **`queue-to hunt-group`** command regardless of success or failure. The **`goto step if queue-fail`** command is provided for handling failure conditions. Otherwise, on success, announcements or other feedback can be applied while the call is in the queue. Since these hunt groups are required to be vector-controlled, announcements are provided by way of vectoring commands and hunt group-specific forced announcements do not apply. If no commands follow a successful queue step, the call is left in the queue with no feedback and vector processing terminates. If no commands follow a failed queue step, the call is dropped. Anytime the end of vector processing is reached without the call being placed in the queue, it is dropped.

route-to number command

Syntax: route-to <number> with cov <y, n> if <unconditionally>

This command is slightly modified from standard usage when used for attendant vectoring and **`unconditionally`** is the only available option. Existing choices allow routing with **`if unconditionally`**, **`digit`**, **`name`**, or **`interflow-qpos`**. Since digit comparison and interflow do not pertain to attendant vectoring, the options are not available. No other changes or attendant specific considerations apply. This command works as it does in standard usage. This command is provided by administration that is defined on the Console Parameters form. Therefore, call processing requirements are not needed.

Syntax: route-to ~r<number>

For incoming calls to the switch, NCR can be activated using the route-to number vector step, where the number field in the vector step has a ~r in the first digit position. This allows for the route-to number vector step to interflow an incoming attendant call to another switch over the PSTN since no trunks are tied up at the redirecting switch.

Branching/programming commands

Attendant Vectoring allows use of several branching/ programming commands, including:

- goto step command *on page 305*
- goto vector command *on page 306*
- stop command on page 306

The following sections detail the syntax that can be used for these commands and any information that is specific to their use in Attendant Vectoring.

goto step command

Syntax: goto step <step #> if time-of-day is <day><hour>:<minute> to <day><hour>:<minute>

This use of the **`goto step`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

Syntax: goto step <step #> if <unconditionally>

This use of the **`goto step`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

Syntax: goto step <step #> if queue-fail and goto vector <vector #> if queue-fail

These vectoring conditionals are available only for attendant vectors. Any time an attempt is made to queue a call and it cannot be queued, these commands can be used to direct vector processing. For attendant vectoring, there is no attempt to determine whether a call can be queued before attempting to do so. Therefore, one of these commands can be used to provide alternate processing when calls cannot be queued. Some examples of why calls can fail to queue are as follows, but this is not a complete list of the causes of failure:

- The queue is full
- The attendant group is in night service and there is no night console
- The individual attendant is not a member of the associated attendant group
- There were invalid multiple queue attempts. See Attendant Vectoring and multiple queueing on page 315 for more information

Failure to queue

The queue failure conditional is set following a queue command that fails to queue the call. It always indicates the result of the most recent queue command. If the failure conditional is set, vector processing is redirected as indicated.

goto vector command

Syntax: goto vector <vector #> if time-of-day is <day><hour>:<minute> to <day><hour>:<minute>

The use of the **`goto step`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

Syntax: goto vector <vector #> if unconditionally

The use of the **`goto step`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

stop command

The use of the **`stop`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

Overview

The Attendant Vectoring capability enables you to use certain vector commands in a non-call center environment. For example applications of Attendant Vectoring see Call Vectoring applications on page 61.

Attendant Vectoring is available in nondistributed attendant environments and distributed attendant environments for IAS and QSIG CAS.

Vector form

The following example shows the Call Vector form with the Attendant Vectoring field enabled.

Call Vector form

```
change vector xxx                                        page 1 of 3
                            CALL VECTOR

    Number: xxx                  Name: ___________________________
Multimedia? n      Attendant Vectoring? y     Meet-me Conf? y          Lock? y
     Basic? n    EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? n    LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? n   Holidays? n

01 ____________
02 ____________
03 ____________
04 ____________
05 ____________
06 ____________
07 ____________
08 ____________
09 ____________
10 ____________
11 ____________
```

The Attendant Vectoring field appears only when Attendant Vectoring is enabled on the Customer Options form. If either Basic Vectoring or Prompting are set to y, the Attendant Vectoring field defaults to n. If Basic Vectoring, Prompting, and Enhanced Conference are not enabled on the Customer Options form, the Attendant Vectoring field defaults to y, and it cannot be changed to n. When the Attendant Vectoring field on the Call Vector form is set to y, that vector is used as an attendant vector.

To associate VDNs and vectors for attendant vectoring, a field on the VDN and the call vectoring forms indicates attendant vectoring. When attendant vectoring is indicated for VDNs and vectors, all call center-associated fields (such as `Skills` and BSR) are not displayed.

Console Parameters form

When Attendant Vectoring is enabled, a field on the Console Parameters form identifies the assigned Attendant Vectoring VDN. The following examples show the Console Parameters forms.

Console Parameters form (Page 1)

```
change console-parameters                                       Page   1 of   4
                              CONSOLE PARAMETERS
         Attendant Group Name: OPERATOR
                          COS: 1                                      COR: 1
       Calls in Queue Warning: 1                   Attendant Lockout? y
        Ext Alert Port (TAAS): 01A1216
                          CAS: none
                                             Night Service Act. Ext.: 195
                 IAS (Branch)? n             IAS Tie Trunk Group No.:
         IAS Att. access Code:                 Alternate FRL Station:
              Backup Alerting? y       DID-LDN Only to LDN Night Ext? n
      Attendant Vectoring VDN: 2000
```

Console Parameters form (Page 2)

```
change console-parameters                                       Page   2 of   4
                              CONSOLE PARAMETERS

TIMING
  Time Reminder on Hold (sec): 30          Return Call Timeout (sec): 30
  Time in Queue Warning (sec): 15

  INCOMING CALL REMINDERS
      No Answer Timeout (sec): 10                      Alerting (sec): 10
                               Secondary Alert on Held Reminder Calls? y

ABBREVIATED DIALING
     List1:                   List2:                   List3: system

                           COMMON SHARED EXTENSIONS
            Starting Extension: 670             Count: 3
```

Console Parameters form (Page 3)

```
change console-parameters                                      Page   3 of   4
                              CONSOLE PARAMETERS

QUEUE PRIORITIES

               Emergency Access: 1
                Assistance Call: 2
                        CO Call: 2
               DID to Attendant: 2
                       Tie Call: 2
            Redirected DID Call: 2
                Redirected Call: 2
                    Return Call: 2
                    Serial Call: 2
    Individual Attendant Access: 2
                Interpositional: 2
       VIP Wakeup Reminder Call: 2
             Miscellaneous Call: 2

       Call-Type Ordering Within Priority Levels? n
```

Console Parameters form (Page 4)

```
change console-parameters                                      Page   4 of   4
                              CONSOLE PARAMETERS

ASSIGNED MEMBERS ( Installed attendant consoles )
              Type       Grp  TN            Type       Grp  TN
           1: principal   1   1          9:
           2:                           10:
           3:                           11:
           4:                           12:
           5:                           13:
           6:                           14:
           7:                           15:
           8:                           16:
```

TN assignments

Just as TN assignment determines to which attendant group calls are terminated, the TN assignment also determines to which VDN the calls are redirected. If a VDN is administered, attendant group calls are redirected to the VDN rather than the attendant group. If a VDN is not assigned, calls terminate to the associated attendant group. How the switch determines which party's TN to use in call scenarios is not changed by attendant vectoring and the VDN for the selected TN still applies.

For example, the selected TN for calls that are covered to an attendant group is the called user's TN, not the calling user's TN. When tenant partitioning is not administered, the system can have only one partition and attendant group. All attendant group calls are directed to attendant group 1. The form to administer TN associations is not accessible, so system-wide console assignments apply. To follow the existing principals of this administration, the attendant vectoring VDN assignment appears on the Console Parameters form when partitioning is turned off. When it is turned on, the field is removed from the console form and the contents are automatically copied to TN 1.

Restrictions

No restrictions apply to attendant and nonattendant vectoring. For example, an attendant VDN can point to a nonattendant vector and vice versa. The same is true for vector commands.

For example, an attendant VDN that points to an attendant vector can have a vector step that routes to another nonattendant VDN. In this case, the call is removed from the queue and treated as though it just entered vector processing rather than as a continuation from one VDN to another. The reverse is also true if a nonattendant VDN is routed to an attendant VDN.

Attendant queue

If attendant vectoring results in putting a call in the attendant queue, it is placed in queue with the priority as administered on the console parameter form. There are no changes made to the attendant priority queue for attendant vectoring. Even when partitioning is turned on and multiple attendant groups exist, all queues have the same priority assignments. Priority queue administration also applies for calls to an individual attendant, by way of the assigned extension.

Hunt group queue

If attendant vectoring results in putting a call in the hunt group queue, it is placed in the queue with the indicated priority. To use this command, the hunt group must be vector controlled.

Redirecting calls to attendant VDNs

Because it is not possible to apply vector commands or specialized administration to specific types of attendant group calls, the following can not be redirected to the attendant VDN:

- Emergency Access. These calls are still sent directly to the attendant group. However, an attendant vectoring VDN can be assigned as the emergency access redirection extension.
- Attendant return calls. These calls are still sent to the original attendant if the original attendant is available or will be placed into the attendant group queue if no attendants are available.
- Serial calls. As with return calls, serial calls are still returned to the original attendant if the original attendant is available and are placed into the attendant queue if no attendants are available.
- VIP Wakeup calls. These reminder calls are still sent directly to the attendant group.
- Call Park time-out. These calls result in a conference (caller, principal, and attendant) and call vectoring does not allow conferenced calls to be vectored.
- Call Transfer time-out. These calls are controlled by the attendant return call timer and are processed as though they are attendant extended calls, in other words, actual attendant return calls.

Night service

There is no additional night service functionality provided for attendant vectoring. Night service routing can be provided using the existing night station service in conjunction with attendant vectoring. All existing night service rules remain in place (for example, night console service supersedes night station service, which supersedes TAAS). Attendant group calls are not redirected to attendant vectoring when the system is in night service unless a night console is available. Otherwise, they continue to be redirected to the applicable night service processing. To achieve attendant vectoring for calls when the system is in night service without a night console, the night station service extensions must be attendant vectoring VDN extensions.

Attendant VDNs

The fact that VDN extensions can be dialed directly or calls can be transferred to VDN extensions is unchanged for attendant VDNs.

Currently, VDN extensions can be assigned to:

Hunt group night destination – An attendant vectoring VDN can be assigned as a hunt group's night destination. Calls to that hunt group when it is in night service are redirected to the VDN and attendant vectoring applies.Hunt group night service does not apply if the hunt group is vector controlled. When `vector?` on the Hunt Group form is y, the `night service destination` field is removed from the form. In order for a hunt group to be available in vectoring for the **`queue-to hunt-group`** command, the hunt group must be vector controlled. The hunt group in the **`route-to`** command could be in night service and the call would then terminate to the indicated night service destination. If the hunt group is accessed via the **`queue-to hunt-group`** command no night service applies.

LDN and trunk night destination – One or all trunk groups can be placed into night service and an attendant vectoring VDN can be assigned as the group's night service destination. If a night destination is assigned for LDN calls, it overrides (for LDN calls) the trunk group's night destination. Either of these destinations can be an attendant vectoring VDN. However, if tenant partitioning is administered and the trunk group night service destination is the attendant group, the call is redirected to the VDN that is associated with the trunk group's TN. If, instead, the night service destination is explicitly assigned to a particular attendant vectoring VDN, it may or may not be the VDN that would have resulted had the night destination been the attendant group.

Tenant night destination – For tenant partitioning, each partition can be assigned a night destination. When tenant partitioning is turned off, local attendant group calls are sent to the LDN night destination. When partitioning is turned on, local attendant seeking calls are sent to the partition's night destination.

Trunk group incoming destination – The incoming destination can be an attendant vectoring VDN except for RLT trunk groups. As in trunk group night service, an assigned incoming destination to an attendant vector could result in the call being sent to a different VDN than if the destination had been assigned to the attendant group.

Last coverage point in a coverage path – An attendant VDN can be assigned as a coverage point. If an Attendant VDN is assigned as a coverage point, it should be the last point in the coverage path.

Abbreviated dialing lists – Attendant VDNs can be assigned to abbreviated dialing lists.

Emergency access redirection – An attendant VDN can be assigned to emergency access redirection. When the attendant's emergency queue overflows or when the attendant group is in night service, all emergency calls are redirected to this VDN. Careful thought should be given to routing these calls off-switch.

QSIG CAS number for attendant group calls – An attendant VDN can be assigned to this number which determines where attendant group calls at a QSIG Branch are processed. This allows local vectoring at a Branch prior to routing the calls to the Main or elsewhere.

Auxiliary data for the following button assignments – In keeping with existing procedures, attendant VDNs will not be denied as auxiliary button data for:

- Facility busy indication. Visual indication of busy or idle status for the associated extension.
- Manual message waiting indication. Lights a message waiting lamp on the station that is associated with the button.
- Manual signaling. Rings the station that is associated with the button.
- Remote message waiting indicator. Message waiting status lamp automatically lights when a LWC message is stored in the system for the associated extension.

Attendant Vectoring and attendant VDNs

When Attendant Vectoring is administered and if an attendant VDN is assigned, attendant group calls are intercepted and sent through vector processing. The attendant VDN can be assigned on the Console Parameters form if tenant partitioning is turned off or on the Tenant form if partitioning is turned on. If an attendant VDN is assigned, the call is redirected to the VDN for vector processing. If a VDN is not assigned, the call is directed to the attendant group. Attendant group calls can only be redirected to attendant VDNs.

Intercept attendant group calls

When calls are placed to the attendant group or become attendant group calls for the reasons listed below, a check is made for an assigned attendant VDN. If an attendant VDN is assigned and either the system is not in night service or the system is in night service and a night console is available, the call is redirected to the VDN for subsequent vector processing. Otherwise, the call is treated with typical attendant group procedures.

The following occurrences can cause a call to become an attendant group call:

- Listed Directory Number (LDN)
- Attendant group in coverage path
- Attendant control of trunk group access
- Calls forwarded to attendant group
- Controlled Restriction
- Dialed attendant access code
- DID/Tie/ISDN intercept treatment
- DID time-out due to Unanswered DID Call Timer expiration

- DID busy treatment
- Security Violation Notification (SVN)
- Multi frequency signaling with attendant group as terminating destination
- CDR buffer full with attendant group as Call Record Handling Option
- Trunk incoming destination is attendant group
- Trunk group night service destination is attendant group
- Hunt group night service destination is attendant group
- Automatic Circuit Assurance (ACA) referral
- VDN routes to the attendant access code.

Vector override always applies to attendant VDNs. The `Allow VDN Override`? field will not be available so **`yes`** is assumed.

Allow override

VDN override always applies to attendant VDNs.

To provide the most flexibility possible, there are no restrictions placed on the vector that is assigned to a VDN. A nonattendant vector can be assigned to an attendant VDN and an attendant vector can be assigned to a nonattendant VDN. Obviously, doing so is not recommended. Assigning an attendant vector to a nonattendant VDN severely restricts processing for basic call vectoring since only limited vectoring commands are available in attendant vectors. Assigning a nonattendant vector to an attendant VDN also severely restricts attendant vectoring since the attendant-specific commands are not available in basic call vectoring. In addition, it removes basic call vectoring information from attendant VDNs. Also, there are no restrictions in vector chaining between attendant and nonattendant vectors (for example, using the **`goto vector`** or **`route-to number`** commands).

Interflow between vectors

When calls interflow from one type of vector processing to another, they are removed from the queue (if applicable) and treated as new calls to vectoring, not continuations of vectoring.

Tenant partitioning assignments apply to attendant VDNs the same as they do for nonattendant VDNs. Therefore, care must be taken that a VDN assignment on the partitioning form has a compatible TN number assigned to the VDN. For example, tenant partition 1 can be assigned a VDN which belongs to tenant partition 2 so long as partition 1’s permissions allow access to partition 2. However, music source selection is based on the tenant partition where the VDN is assigned rather than the partition to which the VDN belongs.

Music source

When music is to be provided for attendant vectored calls, the source that is assigned to the tenant partition of the attendant seeking call is used rather than the source that is assigned to the partition of the VDN.

Attendant Vectoring and multiple queueing

Calls can exist in only one type of queue, which can be an attendant group, and individual attendant, or a hunt queue, and cannot be moved from one queue to another. For example, if a call is queued to the attendant group and a subsequent command attempts to queue the call to an individual attendant or hunt group, it is considered a failed queue attempt.

Restrict queueing to only one type of queue

Once a call is queued to the attendant group, individual attendant, or hunt group, any attempt to queue the call to another type of queue is considered a failed queue attempt.

Multiple attempts to queue to attendant groups or individual attendants are also considered failed queue attempts. For example, if a call is queued to attendant X and a subsequent command attempts to queue the call to attendant Y, the second queue command fails.

Allow multiple priority queueing within hunt queues

Since hunt group queueing is based on the indicated priority, multiple queue attempts are valid. There is no limitation on the number of attempts to queue to a particular hunt group so long as the command changes the priority at which a call is to be queued. For example, a call can be queued at low priority and subsequently requeued at medium and/or high priority. However, a second attempt to queue a call at the same priority for which it was previously queued is considered a failed queue attempt. Hunt group queueing is the functional equivalent to split queueing. As such, calls can be queued to a maximum of three different hunt groups at the same time.

Once a call is queued to a hunt group, any subsequent attempt to queue with a different priority results in the call being requeued with the new priority. Any subsequent attempt to queue with the same priority at which the call is already queued is considered a failed queue attempt.

Allow multiple hunt group queueing

A call can be queued to a maximum of three different hunt groups. Once this maximum is reached, any subsequent attempt to queue a call to a different hunt group is considered a failed queue attempt.

Considerations

The main consideration with Attendant Vectoring is training users to understand that calls placed to an attendant console may not always be answered by a live operator. If users are instructed to dial an attendant VDN, the call could be answered by an attendant, but it may also be covered to the voice mailbox of a night station. Training users to understand these different call routing options is something you should consider before using Attendant Vectoring.

If you use Attendant Vectoring and night service to route calls to a voice mail system, you can also use the Automatic Message Waiting feature to notify after-hours personnel that there are messages in the night service station mailbox by assigning an AMW lamp on one or more backup telephones. When personnel see that there are new messages, they can check those messages after hours and act upon them as needed.

Chapter 15: Holiday Vectoring

Holiday Vectoring enables a set of commands that can be used to write call vectors for calls to be routed on holidays or any days when special processing is required.

This chapter gives you the information you need to use this vectoring option.

This chapter includes the following major topics:

Command set

The following table shows the commands that are available for use in Holiday Vectoring.

Holiday Vectoring command set

Command category	Action taken	Command
Branching/programming		
	Go to a vector step Go to a vector	**`goto step`** **`goto vector`**

Branching/programming commands

Holiday Vectoring allows use of two branching/programming commands, including:

- goto step command *on page 318*
- goto vector command *on page 318*

The following sections detail the syntax that can be used for these commands and any information that is specific to their use in Holiday Vectoring.

goto step command

Syntax: goto step <step #> if holiday in table <table #>

This command directs the call to a specific vector step if the conditions of the call match a holiday that is in the specified Holiday Table.

Syntax: goto step <step #> if holiday not-in table <table #>

This command directs the call to a specific vector step if the conditions of the call do not match any of the holidays that are in the specified Holiday Table.

goto vector command

Syntax: goto vector <vector #> if holiday in table <table #>

This command directs the call to a specific vector if the conditions of the call match a holiday that is in the specified Holiday Table.

Syntax: goto vector <vector #> if holiday not-in table <table #>

This command directs the call to a specific vector if the conditions of the call do not match any of the holidays that are in the specified Holiday Table.

Overview

Holiday Vectoring is an enhancement that simplifies vector writing for holidays. It is designed for customers who need to reroute or provide special handling for date-related calls on a regular basis.

This feature provides the user with the capability to administer ten different Holiday Tables, then use those tables to make vectoring decisions. Each table can contain up to 15 dates or date ranges. All of this can be done in advance to ensure seamless call routing over holidays when staffing is reduced or call centers are closed.

When vector processing encounters a **`goto xxx if holiday in table`** # step, it determines if the current date and time qualifies as a holiday according to the given table. That information is then used to decide whether the goto condition is true or false, and therefore, whether to **`goto`** the given step or vector or not. The date and time match is done at the time that the call is in vector processing. It is done just like time-of-day routing. This means that it is checking the system date and time on the PPN, rather than the local port network time on the EPN.

The Holiday Vectoring feature is not limited to holiday use, but can also be applied to any date-related special processing. For example, vectors can be modified or created to perform special processing during a two-week television promotion or a semiannual sale.

This feature was developed in response to customer needs, especially for some customers who may have as many as 30 bank holidays to administer throughout the year. Holiday Vectoring streamlines vectoring tasks and ensures seamless operation over holiday (or special-event) periods.

Without this feature, call center administrators had to write special vectors for each holiday or other special date-related circumstances, and make sure that these vectors were administered at the appropriate times. In some cases, administrators were required to go to work on holidays just to administer vectors. This feature was developed in response to customer needs, especially for some customers who may have as many as 30 bank holidays to administer throughout the year.

Administering Holiday Vectoring

This section gives you step-by-step instructions on setting up Holiday Tables and writing vectors to include Holiday Vectoring.

Enabling Holiday Vectoring

The Holiday Vectoring customer option can be enabled if:

- The switch software version is R9.1 or greater, and
- Either Vectoring (Basic) or Attendant Vectoring is enabled.

On the Customer Options Form, the Vectoring (Holidays) field should be set to y. If the feature is not enabled, contact your Avaya customer support or authorized representative to have the feature enabled.

Setting up a Holiday Table

This section describes how to set up a Holiday Table before adding to a vector.

Holiday Table command syntax

This section describes the syntax of each Holiday Vectoring command.

Syntax: change holiday-table x

This command allows you to change the entries in a Holiday Table.

To create a new Holiday Table, you must use the change command and give the number of a blank table. For example, change holiday-table 9, where table 9 has not been used to define holidays.

Syntax: display holiday-table x

This command allows you to display the entries in a Holiday Table.

Syntax: list holiday-table

This command lists all of the Holiday Tables.

Syntax: list usage holiday-table x

This command lists all vector steps that refer to the selected Holiday Table.

Using the Holiday Table commands

After ensuring that Holiday Vectoring is enabled on the Customer Options form, enter **`change holiday-table 1`**.

On the Holiday Table Form, which is shown in the following example, enter the holiday information.

Setting up a Holiday Table

```
change holiday-table 1                                      page 1 of 1
                                HOLIDAY TABLE

    Number: 1             Name: Bank Holidays

          START                  END
    Month Day Hour Min    Month Day Hour Min     Description
    12    24              12    31               Christmas
    01    01  00   00     01    01  10   00      New Year's Day
```

Note:

When using a range of dates, the end date must be greater than the start date. Ranges must be within one calendar year. In the example above, two entries were made, one for each calendar year.

The Holiday Table Form can be used for entering individual holidays or holiday ranges. The following rules apply to entering dates on this form:

- If a day is entered, the corresponding month must be entered.
- If a month is entered, the corresponding day must be entered.
- If an hour is entered, the corresponding minute must be entered.
- If a minute is entered, the corresponding hour must be entered.
- If an hour and minute is entered, the corresponding month and day must be entered.
- If a month and day is entered, the corresponding hour and minute is not required.
- If an end month and day is entered, the corresponding start month and day must be entered.
- If a start month and day is entered, the corresponding end month and day is not required.
- To enter an individual holiday, enter a start month and day, but do not enter an end month and day.

- To enter a holiday range, enter both a start month and day and an end month and day.
- The start month, day, hour, and minute must be less than or equal to the end month, day, hour, minute.
- The description field is an alpha-numeric field that is used for identification.

After creating a holiday table, use the **`display holiday-table`** command to view the entries. To list all of the holiday tables, use the **`list holiday-table`** command, as shown in the following example.

Listing the Holiday Tables

```
list holiday-table
                                HOLIDAY TABLES

                          Table Number      Name
                              01            Business Holidays
                              02            Annual Promotion Dates
                              03            Summer Special
                              04
                              05
                              06
                              07
                              08
                              09
                              10
```

Changing vector processing for holidays

After administering the holiday tables, add or change vector processing for those holidays.

On the command line, enter **`change vector`** ***`x`*** (where ***x*** is the vector number). The Call Vector form contains a display-only field that indicates that Holiday Vectoring is enabled. On the Call Vector form, customers can enter a new goto conditional for the holidays.

When Holiday Vectoring is optioned, a field on the Vector form identifies if the vector on which you are currently working is a Holiday Vectoring vector, as shown in the following example.

Call Vector form

```
change vector x                                                    page 1 of 3
                            CALL VECTOR

    Number: xxx      Name: ___________________________
Multimedia? n      Attendant Vectoring? n    Meet-me Conf? n           Lock? y
     Basic? y   EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? y   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? n   Holidays? y

01 ____________
02 ____________
03 ____________
04 ____________
05 ____________
06 ____________
07 ____________
08 ____________
09 ____________
10 ____________
11 ____________
```

The Holiday Vectoring field is a display-only field and appears only when Holiday Vectoring is enabled on the Customer Options form. If either Basic Vectoring or Attendant Vectoring are set to y, then the Holiday Vectoring field can be set to y.

The following examples use **goto** commands to route calls for holidays:.

Holiday Vectoring example 1

```
 change vector 1                                                 Page 1 of 3
                        CALL VECTOR

    Number: 1        Name: In Germany
Multimedia? n      Attendant Vectoring? n    Meet-me Conf? n           Lock? y
     Basic? y   EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? y   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? n   Holidays? y

01 goto         vector 2  if holiday         in    table 1
02 route-to     number 123456789       with cov n if unconditionally
03
04
05
06
07
08
09
10
11
```

Holiday Vectoring example 2

```
change vector 3                                              Page 1 of 3
                    CALL VECTOR

    Number: 3        Name: In Ireland
Multimedia? n      Attendant Vectoring? n    Meet-me Conf? n           Lock? y
     Basic? y   EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? y   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? n   Holidays? y

01 goto         step   2  if holiday         in    table 2
02 route-to     number 45678          with cov n if unconditionally
03 stop
04 announcement 2721
05
06
07
08
09
10
11
```

After you have assigned Holiday Tables to several vectors, you can use the **`list usage holiday-table`** command, as shown in the following example, to display which vectors and vector steps are using the selected Holiday Table.

List of Holiday Table use in vectors

```
list usage holiday-table
                              LIST USAGE REPORT

     Used By
     Vector                Vector Number 1                  Step 1
     Vector                Vector Number 3                  Step 1
```

Holiday Vectoring considerations

Consider the following when administering Holiday Vectoring:

- Administration of Holiday Tables is supported only on the switch and cannot be changed using adjunct vectoring tools.
- Holiday Vectoring is only available when Vectoring (Basic) or Attendant Vectoring is enabled.
- There is no validation that verifies the consistency among the 15 holidays in any table. If the same holiday is entered twice, the system stops checking with the first entry that is found.
- With holidays that are ranges of dates, the ranges could overlap. When a call is in vector processing, the holidays are checked from top to bottom on the table and the check stops if a match is found. Even though there might be multiple entries that would match, the check stops at the first match.
- There is a validation that the day of the month that is entered is valid with the given month. Specifically, if the month is April, June, September, or November, then the date must be a number between 1 and 30. If the month is January, March, May, July, August, October, or December, then the date can be a number between 1 and 31. If the month is February, then a the date can be a number between 1 and 29.

  Note:

  The year is not checked in holiday vector processing. This allows the same holidays to be used year-to-year when the holiday is on a fixed date. For holidays where the date changes from year-to-year, the holiday tables must be readministered.
- When disabling the Holiday Vectoring feature (changing the value of the `Vectoring (Holidays)` field from `y` to `n` on the Customer Options form), the vectors are checked for any **`goto...if holiday`** steps. If any of these steps are found, an error message is displayed, and the change is not allowed. The customer must remove those vector steps first before the feature can be disabled.

Chapter 16: Meet-me Conference

The Meet-me Conference feature allows you to set up a dial-in conference of up to six parties. The Meet-me Conference feature uses Call Vectoring to process the setup of the conference call.

Meet-me Conference can be optionally assigned to require an access code. If an access code is assigned, and if the vector is programmed to expect an access code, each user dialing in to the conference call must enter the correct access code to be added to the call.

The Meet-me Conference extension can be dialed by any internal or remote access users, and by external parties if the extension number is part of the customer's DID block.

This chapter includes the following sections:

Command set

The following table lists the commands associated with Meet-me Conference.

Meet-me Conference command set

Command category	Action taken	Command
Information collection		
	Collect information from the calling party.	**`collect digits`**
Treatment		
	Play an announcement.	**`announcement`**
	Play a busy tone and stop vector processing.	**`busy`**
	Disconnect the call.	**`disconnect`**
	Delay with audible feedback of silence, ringback, system music, or alternate audio or music source.	**`wait-time`**
Routing		
	Route to the appropriate meet-me conference and stop vector processing.	**`route-to`**
Branching/Programming		
	Go to a vector step.	**`goto step`**
	Go to another vector.	**`goto vector`**
	Stop vector processing.	**`stop`**

Information collection commands

Meet-me Conference uses the following information collection commands:

- collect command on page 329

The following section details the syntax that can be used for this command and any information that is specific to the Meet-me Conference feature.

collect command

Syntax: collect 6 digits after announcement <extension>

When the `Meet-me Conf` field is enabled, the **`collect`** vector step has been modified to collect the next six digits and use those digits as the access code for a Meet-me Conference call. Though not required, the digits can be collected after a recorded announcement.

Treatment commands

Attendant Vectoring allows use of several treatment commands, including:

- announcement command on page 329
- busy command on page 329
- disconnect command on page 330
- wait-time command on page 330

The following sections detail the syntax that can be used for these commands and any information that is specific to the Meet-me Conference feature.

announcement command

Syntax: announcement <extension>

The usage for the **`announcement`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

busy command

Syntax: busy

The usage for the **`busy`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

disconnect command

Syntax: disconnect after announcement <extension>

The usage for the **disconnect** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

wait-time command

Syntax: wait-time <time> secs hearing <silence, ringback, music>

The usage for the **wait-time** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

Routing commands

Meet-me Conference uses the following routing command:

- route-to meetme command on page 330

The following section details the syntax that can be used for this command and any information that is specific to the Meet-me Conference feature.

route-to meetme command

Syntax: route-to meetme

The **route-to** vector step uses the condition **meetme** only for the Meet-me Conference feature. When successful, this condition adds the caller to the Meet-me Conference call and all parties on the call hear an “entry” tone to signify that another caller has joined the conference. This condition is valid when the caller has entered the correct access code and there are not already six parties on the call.

If the **route to meetme** step ever fails, vector processing stops and the caller hears busy tone.

Branching/programming commands

Meet-me Conference uses several branching/ programming commands, including:

- goto step command *on page 331*
- stop command on page 331

The following sections detail the syntax that can be used for these commands and any information that is specific to their use in Attendant Vectoring.

goto step command

Syntax: goto step <step #> if meet-me-idle

Syntax: goto step <step #> if meet-me-full

The **`goto step`** vector step has two conditions used for the Meet-me Conference feature:

- meet-me-idle
- meet-me-full

The **`meet-me-idle`** condition routes the first caller accessing a Meet-me Conference to the conference call. An announcement step saying they are the first party to access the call can be given to the caller.

The **`meet-me-full`** condition is used when the Meet-me Conference already has the maximum of six parties on the call.

Syntax: goto step <step #> if digits = meet-me-access

The **`goto step`** vector step supports the option, **`meet-me access`**, for the **`digits`** condition to verify that the access code is valid. If the access code entered by the caller equals the access code administered for the VDN, vector processing continues.

stop command

The use of the **`stop`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

Administering Meet-me Conference

This section contains the following information:

Activating the Meet-me Conference feature

Meet-me Conference is available for all switch models that support the R11 call processing software.

To enable the Meet-me Conference feature:

- The G3 Version field of the Customer Options screen must be set to V11 or later.
- The Enhanced Conferencing field of the Customer Options screen must be enabled. This feature has an RTU cost and must be enabled through the License File process.

Creating a Meet-me Conference VDN

To create a Meet-me Conference VDN (using example VDN 36090):

1. Enter:

   `add vdn 36090`

   The system displays the `VECTOR DIRECTORY NUMBER` screen:

```
add vdn 36090                                                  Page   1 of   2   SPE A
                            VECTOR DIRECTORY NUMBER

                             Extension: 36090
                                  Name: Meet-me VDN
                         Vector Number: 90

                    Meet-me Conference? y

                                   COR: 1
                                    TN: 1
```

2. Enter a name, a vector number, and enter **y** in the `Meet-me Conference` field.
3. Press **NEXTPAGE** to display page 2.

   The system displays page 2 of the `VECTOR DIRECTORY NUMBER` screen:

```
add vdn 36090                                          Page   2 of   2   SPE A
                            VECTOR DIRECTORY NUMBER

                         MEET-ME CONFERENCE PARAMETERS
       Conference Access Code: 937821
        Conference Controller: 80378
```

4. Enter a conference access code. If you do not want an access code, leave the field blank. Once an access code is assigned, an asterisk displays in this field for subsequent change, display, or remove operations by all users except the “init” superuser login.

   SECURITY ALERT:
   You should always assign an access code to a Meet-me Conference VDN.

5. Enter a conference controller extension. If an extension number is entered, a user at that extension can change the access code for the Meet-me Conference VDN using a feature access code. If this field is blank, only a station user that is assigned with console permissions can change the access code for the Meet-me Conference VDN using a feature access code. In addition, remote access users can change a Meet-me Conference access code using the feature access code.
6. Press **ENTER** to submit the VDN.

Creating a Meet-me Conference vector

To create a Meet-me Conference vector (using example vector number 90):

1. Enter:

   `change vector 90`

   The system displays the `CALL VECTOR` screen.
2. Enter **y** in the `Meet-me Conf` field. This designates the vector as a Meet-me Conference vector.

3. Create a vector as shown in the following example:

```
change vector 90                                                 Page   1 of   3   SPE A
                                  CALL VECTOR

    Number: 90               Name: Meet-me Vec
Multimedia? n      Attendant Vectoring? n    Meet-me Conf? y           Lock? y
     Basic? y   EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? y   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? n   Holidays? n

01 collect      6    digits after announcement 12340
02 goto         step   6   if digits           =      meet-me-access
03 collect      6    digits after announcement 12341
04 goto         step   6   if digits           =      meet-me-access
05 disconnect   after announcement 12342
06 goto         step   11  if meet-me-idle
07 goto         step   14  if meet-me-full
08 announcement 12343
09 route-to     meetme
10 stop
11 announcement 12344
```

```
change vector 90                                                 Page   2 of   3   SPE A
                                  CALL VECTOR

12 route-to     meetme
13 stop
14 disconnect   after announcement 12345
15 stop
16
17
18
19
20
21
22
```

4. Press **ENTER** to submit the vector.

Interactions

The following are administration interactions for Meet-me Conference.

General

Both Attendant Vectoring and Meet-me Conference cannot be enabled at the same time.

If Enhanced Conferencing is enabled, but no other vectoring customer options are enabled, only Meet-me Conference vectors can be assigned.

A non Meet-me Conference vector cannot be assigned to a Meet-me Conference VDN and a Meet-me Conference vector cannot be assigned to a non Meet-me Conference VDN.

There will be no restrictions in vector chaining between Meet-me Conference and non Meet-me Conference vectors (for example, using the **`goto vector`** or **`route-to number`** commands). When calls interflow from one type of vector processing to another, they will be removed from any queue (if applicable) and treated as new calls to vectoring, not a continuation of vectoring.

Call Detail Recording

As parties join a Meet-me Conference, a call record is created if required by system administration. If a record is required, the called party will be the Meet-me Conference VDN number and the duration will be the length of time that the party was included in the call. There will be an individual record for each party that will be output when the party drops from the call. One option that will record all calls to Meet-me Conference VDNs is to activate the Intra-switch CDR feature and populate all the Meet-me Conference VDN numbers in the system.

If the Intra-switch CDR feature is used with the Meet-me Conference VDNs, the condition code should be set to “C” for all call records as is done with traditional conference calls when Intra-switch CDR is active.

If Intra-switch CDR feature is not active for Meet-me Conference VDNs, the creation and contents of call records will depend on the trunk group translations for external callers to the Meet-me Conference. Internal callers to the Meet-me Conference will not generate any records if the Intra-switch CDR feature is not active for either the Meet-me Conference VDN or the calling extension.

Changing vector types

To change a Meet-me Conference vector to a non Meet-me Conference vector, the administrator must first remove all vector steps. To change a non Meet-me Conference vector to a Meet-me Conference vector, the administrator must first remove all vector steps. If either of these conditions exist, a warning message displays that states “VDNs currently assigned to this vector may not operate as expected.” The next time the administrator tries to submit a change to the Meet-me Conference VDN, they would be forced to assign the VDN to a Meet-me Conference vector.

Direct Inward Dialing (DID)

If the VDN extension is part of the customer’s DID block, external users will be able to access the conference VDN. If the VDN extension is not part of the customer’s DID block, only internal callers on the customer’s network (including DCS or QSIG) or remote access callers can access the conference VDN.

Disabling Enhanced Conferencing

If Meet-me Conference VDNs are assigned when disabling the Enhanced Conferencing option, the change is not allowed and the message “Must first remove all Meet-me Conf VDNs and vectors” is displayed. The administrator must remove those VDNs and vectors before the option can be disabled.

Removing stations

A station that is administered as a controlling station for a Meet-me Conference VDN cannot be removed without first removing the assignment on the VDN. The message “Must first remove as conference controller on VDN form” is displayed.

Security issues

The Meet-me Conference feature is a potential security problem. If Meet-me Conference VDNs are assigned without access codes, hackers could tie up Meet-me Conference facilities, keeping others from conducting legitimate business, and could potentially access the switch and use the switch to make unauthorized calls. Therefore, we should recommend that all Meet-me Conference VDNs have access codes that are known only to administrators and users on a need to know basis. We should also recommend that access codes be changed on a regular basis to reduce the risk of unauthorized access to the switch.

If a user tries to change the access code of a Meet-me Conference and is unsuccessful, or if a user tries to access a Meet-me Conference and uses an invalid access code, a meet-me event is logged. See Tracking unexpected events on page 512.

Capacity issues

Meet-me Conference calls count towards the maximum number of 3-way and 6-way conference calls.

Users cannot add more parties to a conference call once the system maximum is reached.

For Category A, the number of Meet-me Conference VDNs is a subset of the total number of VDNs allowed in the system.

For Category B, the total number of VDNs and vectors is doubled from the normal limit if both Call Vectoring and Enhanced Conferencing are enabled. However, the maximum number of VDNs and vectors available for call center applications is unchanged.

Meet-me Conference call processing scenario

Joe Davis has a sales review scheduled with four associates located in different cities. He has reserved Meet-me Conference telephone number 865-253-6090. In switch administration, this number has been assigned to vector 90. See the following screen.

```
add vdn 36090                                                  Page   1 of   2   SPE A
                            VECTOR DIRECTORY NUMBER

                             Extension: 36090
                                  Name: Meet-me VDN
                         Vector Number: 90

                    Meet-me Conference? y

                                   COR: 1
                                    TN: 1
```

VDN 36090 is administered with an access code of 835944. See the following screen.

```
add vdn 36090                                               Page   2 of   2   SPE A
                            VECTOR DIRECTORY NUMBER

                         MEET-ME CONFERENCE PARAMETERS
       Conference Access Code: 835944
        Conference Controller:
```

When each associate calls the Meet-me Conference telephone number, the following vector processing occurs:

```
change vector 90                                         Page   1 of   3   SPE A
                                  CALL VECTOR

    Number: 90               Name: Meet-me Vec
                    Attendant Vectoring? n    Meet-me Conf? y           Lock? y
     Basic? y   EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? y   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? n   Holidays? n

01 collect      6    digits after announcement 12340
02 goto         step   6   if digits           =      meet-me-access
03 collect      6    digits after announcement 12341
04 goto         step   6   if digits           =      meet-me-access
05 disconnect   after announcement 12342
06 goto         step   11  if meet-me-idle
07 goto         step   14  if meet-me-full
08 announcement 12343
09 route-to     meetme
10 stop
11 announcement 12344
```

```
change vector 90                                         Page   2 of   3   SPE A
                                  CALL VECTOR

12 route-to     meetme
13 stop
14 disconnect   after announcement 12345
15 stop
16
17
18
19
20
21
22
```

Each caller hears announcement 12340, which says something similar to “Welcome to the Meet-me Conferencing service. Enter your conference access code.” Each caller enters the access code 835944.

The **`collect`** vector step 1 collects the access code digits. If the access code is valid, the vector processing continues with vector step 6. If the access code is invalid, the vector processing continues with vector step 3, which plays announcement 12341. Announcement 12341 says something similar to “This access code is invalid. Please enter the access code again.” If the caller enters the wrong access code again, the vector processing continues with vector step 5, which plays announcement 12342. Announcement 12342 says something similar to “This access code is invalid. Please contact the conference call coordinator to make sure you have the correct conference telephone number and access code. Good-bye.”

Vector step 6 is only valid for the first caller into the Meet-me Conference. The `meet-me-idle` condition routes the first caller to announcement 12344 (vector step 11). The recorded announcement says something similar to “You are the first party to join the call.” The caller is then routed to the Meet-me Conference call by vector step 12 and vector processing stops.

Vector step 7 is used when the Meet-me Conference already has the maximum of six parties on the call. The `meet-me-full` condition disconnects the caller after playing announcement 12345 (vector step 14). The recorded announcement says something similar to “This Meet-me Conference is filled to capacity. Please contact the conference call coordinator for assistance. Good-bye.”

If a caller enters the correct access code, is not the first caller, and the conference call is not full, vector processing continues with vector step 8, which plays announcement 12343. The announcement says something similar to “Your conference call is already in progress.” The caller is then routed to the Meet-me Conference call by vector step 9 and vector processing stops. As each caller enters the conference call, all parties on the call will hear an “entry” tone.

When the conference call is over and callers drop out of the conference call, any remaining parties on the call will hear an “exit” tone.

Chapter 17: Expert Agent Selection

This chapter describes EAS, and provides examples that show how EAS is implemented. The chapter also discusses EAS upgrades. Topics include:

Special EAS-related considerations

When you implement the EAS feature, be aware of the following special considerations:

- With EAS, skill hunt groups replace splits. You cannot administer both skills and splits on the same switch. All ACD hunt groups must be administered as either splits or skills. If EAS is optioned, all ACD hunt groups are skill hunt groups.
- With EAS, all skill hunt groups except for AUDIX hunt groups must be vector controlled.
- With EAS, non-ACD hunt groups are allowed, but they cannot be vector controlled.
- Agent Login IDs are extensions in the dial plan, and they decrease the total number of stations that can be administered.
- With EAS, agents have a different login procedure and a single set of work mode buttons, regardless of the number of skills that are assigned to the agents.
- Skill hunt groups can distribute a call to the most-idle agent (UCD) or to the most-idle agent with the highest skill level for that skill (EAD). In either of these cases, the call can route to the most-idle agent for the specified skill, or to the most-idle agent in all of the skills. Direct Department Call (DDC) distribution is not allowed for skill hunt groups.
- With either UCD or EAD distribution, the system can be administered to deliver calls based either on greatest need or agent skill level. This is the Call Handling Preference that is administered on the Agent LoginID form. When calls are in the queue, greatest need delivers the highest priority oldest call waiting for any of the agent's skills. With skill level administration, the system delivers the highest priority oldest call waiting for the agent's highest level skill with calls in the queue.
- The EAS-PHD customer option adds additional capabilities to the basic EAS capabilities.
  - It increases the number of skills an agent can log in to from 4 to 20
  - It increases the number of agent skill priority levels from 2 to 16

For information on converting a call center to EAS, see Converting a Call Center to EAS on page 617.

Expert Agent Selection (EAS) terminology

The following terms have special significance in the EAS environment.

EAS terminology

Agent skill	The type of call that a particular agent can handle. With EAS, an agent can be assigned up to four skills each, with a primary (level 1) or secondary (level 2) skill level. With EAS-PHD, an agent can be assigned as many as 20 skills.
Caller needs	The reasons why customers call your call center. Caller needs are determined by the VDN number that the caller dialed, by Call Prompting, or by Automatic Number Identification (ANI) database lookup. You define caller requirements in the vector in order to route calls to an ACD agent with particular skills to match the needs of the caller. These caller needs, which translate to skills, become active for an ACD call whenever a queue to the main skill or check backup skill vector command is executed and the threshold condition is met.
Skill	A specific caller or business need of your call center. You define your skills based on the needs of your customers and your call center. You specify skills by skill numbers, which are assigned to agents and are referenced in vectors to match caller needs with an agent who is skilled to handle those needs. When configuring your call center for skills, a particular skill number always has the same meaning, whether it is an agent skill, VDN skill, or skill hunt group.
Skill hunt group	Calls are routed to specific skill hunt groups that are usually based on caller needs. Agents are not assigned to a skill group; instead, they are assigned specific skills that become active when they log in.
Skill level	For each agent skill, a skill level may be assigned. With EAS-PHD, skill levels can range from 1 to 16, with 1 being the highest skill level (also known as the highest-priority skill). Without EAS-PHD, skill levels may be defined as primary (level 1) or secondary (level 2), with the primary being the highest-priority skill. When calls are queued for more than one of the agent’s skills and the agent’s call-handling preference is by skill level, the agent receives the oldest call waiting for the agent’s highest level skill. If an agent’s call-handling preference is by greatest need, then the agent receives the highest-priority, oldest call waiting for any of that agent’s skills, regardless of skill level.
Top agent	An agent in a given skill who has the skill assigned as top skill.

EAS terminology (continued)

Top skill	For EAS-PHD, an agent's first-administered, highest-priority skill. For EAS, an agent's first-administered primary skill (or first-administered secondary skill if the agent has no primary skill assigned). With call-handling preference by skill level, this is the skill for which the agent is most likely to receive a call.
VDN skill preference	Up to three skills can be assigned to a VDN. Calls use VDN skills for routing based on the preferences that you specify in the vector. VDN skill preferences are referred to in the vector as 1st, 2nd, and 3rd.

What is Expert Agent Selection (EAS)?

Expert Agent Selection (EAS) helps call center managers to provide the best possible telephone service to callers by matching the needs of the callers with the skills or talents of the agents. Caller needs and agent skills are matched using Call Vectoring. All the Call Vectoring features described in this guide can be used with EAS.

Matching the call to an agent with the appropriate skills reduces transfers and call-holding time. Accordingly, customer satisfaction is increased. Also, since an entire agent group need not be trained at the same time for the same skills, employee satisfaction is increased.

In addition to matching the skills that are required for a call to an agent with one of those skills, EAS provides other capabilities:

- Logical Agent associates hardware (the telephone) with an agent only when the agent is logged in. While the agent is logged in, calls to the Agent Login ID are directed to the agent. See Logical Agent capability on page 362 for more details.
- Direct Agent Calling allows a user to call a particular agent and have the call treated as an ACD call. See Direct Agent calling on page 351 for more details.

Most EAS administration can be completed before you activate it, thus minimizing the down time for upgrading to EAS.

EAS requires ACD and Call Vectoring. All of the existing ACD features and Call Vectoring capabilities can be used within EAS applications.

As with Call Vectoring calls, EAS calls are directed to VDNs, which in turn point to vectors. However, unlike Basic Call Vectoring, skills can be assigned in EAS to VDNs, or they can be associated with vector steps to represent caller needs. As for Call Vectoring calls, EAS calls are queued to ACD hunt groups. However, with EAS enabled, ACD hunt groups are called “skill hunt groups” instead of splits.

Skill hunt groups deliver calls to EAS agents. Agent skills are administered on the Agent LoginID form.

Note:

These are the same login IDs that are used by Avaya Call Management System (CMS) and Basic Call Management System (BCMS).

Logical Agent implies that telephones are no longer preassigned to hunt groups. When the agent logs, the telephone becomes associated with all of the skill hunt groups that are assigned to that Agent Login ID.

With EAS optioned and enabled, ACD calls can also be directed to a particular agent, instead of to the skill hunt group, by using the Direct Agent Calling feature. The Direct Agent call is treated like an ACD call, but it waits in queue for a specific agent to become available. Direct Agent calls have a higher priority than skill hunt group calls.

EAS benefits

Because you can match caller needs to an agent who has the appropriate skills to handle the call, your call center can achieve the following:

- Maximum profitability.
- Greater customer satisfaction because the caller reaches, on the first call, an agent with the necessary skills to handle the call.
- Greater responsiveness to customer needs because you can base call distribution on either skill level or greatest need.
- Improved agent performance and satisfaction because agents handle calls they are most familiar and most comfortable with.
- Improved agent performance because supervisors have the option to have agents handle calls based on either skill level or greatest need. For agents, it offers an opportunity to learn new skills.
- Ability to track the number of calls that are handled by particular skills from the VDN perspective. You can see whether vectors are performing as expected.

Skill-based call distribution

With EAS, call distribution is based on agent skills. Caller needs are determined by the VDN called or by voice prompting.

An agent who has at least one of the skills that a caller requires is selected to handle the call. You assign skills and skill levels to agents to determine which types of calls go to which agents and to determine the order in which agents serve waiting calls.

Greatest need call distribution

With EAS, you have the option of basing call distribution on greatest need instead of skill level. You can distribute the highest-priority, oldest call waiting to an agent with an appropriate skill, even if that skill is not the agent's highest-priority skill.

Percent allocation call distribution

Percent allocation enables you to assign a percentage of an agent's time to each of the agent's assigned skills, to comprise a total of 100% of the agent's staffed tome. Percent allocation then selects the call that is the best match for an agent's administered skill percentages.

Percent allocation is available with Avaya Business Advocate. For more information, see *Avaya Business Advocate User Guide*, 585-210-711.

ACD queuing and vector commands

ACD queuing and the vector commands **`queue to skill`** and **`check skill`** are used to route a call to an agent with the appropriate skill to handle the call.

EAS-PHD — 20 skills/16 skill levels

EAS-PHD is a feature that allows an agent to be assigned to as many as 20 skills. For each skill, one of the 16 skill levels can be assigned, with 1 being the highest skill level and 16 being the lowest skill level.

If calls are waiting for some of the agent's skills and the agent's call-handling preference is by skill level, the agent receives the call that requires the agent's highest-priority skill. For an agent, the first-administered, highest-priority skill is known as the agent's "top skill." The top skill represents the skill for which the agent is most likely to receive a call.

If an agent's call-handling preference is by greatest need, the top skill is not useful, because the agent receives the highest-priority, oldest call waiting that requires any of the agent's skills, regardless of skill level.

Switch administration for the EAS feature

Before activating EAS in your call center, you will need to complete the appropriate forms on your Avaya switch, as outlined in the following sections.

The “Functions and Examples” section gives details about the administration of the EAS feature using many of these forms.

See *Avaya MultiVantage Call Center Software – Guide to ACD Call Center*, 555-230-716 for more information on the forms listed below.

The following table lists the forms used to administer EAS.

EAS administration forms

Form	Use
System-Parameters Customer-Options	The `Expert Agent Selection Enabled?` field on this form changes to **y** when EAS is installed. If you purchased EAS-PHD, the `Expert Agent Selection-Preference Handling Distribution (EAS-PHD) Enabled?` field changes to **y**.
Dial plan	Use this form to change the dial plan. It is recommended that login IDs start with a unique digit in the dial plan (for example, 5111, 5123, 5432). It is preferable to dedicate a block of numbers for login IDs. If your login IDs do not have the same first digit and the login IDs are four digits long, consider changing to a 5-digit number for login IDs. This may require a modification to the CMS login ID if the current ID is not a valid extension number or cannot be made available in the switch dial plan. Agent login IDs must be different from assigned telephone extensions.
VDN	Use this form to add or change VDNs and to designate skill preferences.
Vector	Use this form to change vectors.
Hunt Group	Use this form to add or change skill hunt groups. The `Skill?`, `ACD?` and `Vector?` fields must be all **y** or all **n**. Hunt group types should be either **UCD** or **EAD**. You cannot administer agents on this form when EAS is enabled.

EAS administration forms ***(continued)***

Form	Use
Agent Login ID	Use this form to add or change agent login IDs and skill assignments. If you add or change skills on the switch, the agent must log out and then log in again before the changes take effect. You must use the Agent Login ID form to select call-handling preferences for agent login IDs. The `Call Handling Preference` field must be set to either **skill level** or **greatest need**. The default is skill level. You also may enter a direct agent skill number in the `Direct Agent Skill` field. The skill entered in this field must be one of the agent's administered skills or the field is left blank. If no direct agent skill is administered and the agent receives a direct agent call, the call is delivered to the agent's first-administered, highest-level skill.
Station	Only a single set of work mode buttons is needed with EAS. Use this form to remove additional sets of buttons if you are administering agents in multiple splits.

Identifying caller needs

Caller needs for a particular call can be identified by any of the following methods:

- Interpreting information that is passed from the network in the form of DNIS digits or ISDN messages.
- Processing Call Prompting digits, digits entered at a Voice Response Unit (VRU), or CINFO digits that are forwarded by the network.
- Using ASAI or a VRU such as the Conversant in a host database lookup.

To show how a call center manager might match caller needs and agent skills (which can be viewed as capabilities needed from the caller's perspective), assume that a call center receives inbound calls from automobile club members who speak Spanish or English. The callers in this case either need to plan a vacation route or have trouble with their car and are calling for assistance. The following table provides example associations between caller needs and agent capabilities.

Example of caller need-to-agent skill matching

Caller need	Capability needed
Tourist information	Knowledge of the region
To speak Spanish	Bilingual
Emergency assistance	Handle stressful callers
Tow truck	Access to dispatch systems)

The following list looks at the call center manager's strategy in matching the caller needs to the capabilities of the agent:

- Tourist information/knowledge of the region

  Travelers may need information while traveling or regarding a future trip. All assigned agents can provide this information.

- To speak Spanish/bilingual

  Separate numbers are published and used as part of Spanish membership information, or Call Prompting is used after a general number is dialed.

- Emergency assistance/handle stressful callers

  Separate emergency road service numbers are published and used, or Call Prompting is used after a general number is dialed. For example, a number is provided for towing.

Note that the call center chose to implement Call Prompting to identify Spanish-speaking callers and callers who require emergency assistance. This allows for quicker and more specialized treatment and therefore better satisfies the caller's needs.

In addition, some customers might prefer to speak to the agent that he or she spoke to on a previous call. To accommodate this request, a call center manager can implement Direct Inward Dialing (DID) at the call center. Also, direct agent calling can be used to direct a call to a specific agent.

The following sections explain further how caller needs are identified.

DNIS/ISDN called party

A set of DNIS digits can be interpreted as a VDN. The following table presents four services and their corresponding telephone number including DNIS digits that might be provided to the caller.

Examples of services and corresponding DNIS digits

Service	Telephone number	Corresponding DNIS
Emergency road service (English)	800-765-1111	6001
Emergency road service (Spanish)	800-765-2222	6002
Route planning (English)	800-765-3333	6003
Route planning (Spanish)	800-765-4444	6004
General (Call Prompting)	800-765-5555	6005

Note:

DNIS digits must be extensions that are reflected in the dial plan.

Call Prompting/VRU Digits/CINFO digits

The Call Prompting/VRU/CINFO digits are entered by the caller in response to any recorded question about a caller's needs, or in the case of CINFO ced or cdpd digits, are provided by the call center host computer. For example, a hotline for a product may request that a product code be entered, or a travel service may request a 2-digit state code to indicate the state to which the caller would like to travel. The following table provides a prompt that encourages the caller to enter the appropriate Call Prompting digit for the needed service from the automobile club.

Example of a prompt for entering Call Prompting digits

For emergency road service, dial 1.
Para asistencia con su automovil, marque el dos.
For travel route directions, dial 3.
Para informacion sobre rutas, marque el cuatro."

Host database lookup

A host database lookup uses DNIS and ANI (calling party's number) to determine what skills are required or even the agent desired. For example, the database may show that the caller speaks Spanish and has been working with Agent 1367. To access host information, either ASAI or a VRU in conjunction with a **`converse-on skill`** step is used.

Direct Agent calling

Direct agent calling allows a call to a specific ACD agent to be treated as an ACD call. Zip-tone answer, ACW, and other ACD features can be used with Direct Agent calls.

If an agent is logged in but is not available, the call queues for that agent. If the agent is not logged in, the call follows the agent's coverage path.

EAS Direct Agent calling is accomplished by dialing the login with the proper class of restriction (COR) settings. Both the caller (that is, trunk, VND, or station) and the agent must have the Direct Agent COR settings.

Customers might call an agent directly using Direct Inward Dialing (DID) if the agent's login ID is a published number, or customers might dial a toll-free number and be prompted for the agent's login ID extension. Vectors can be designed to handle the Call Prompting function.

Direct agent calling

Note:

Direct Agent calling requires CallVisor Adjunct-Switch Application Interface (ASAI) or EAS. Both originating and called party Class of Restrictions (CORs) must be set to allow Direct Agent Dialing.

Direct Agent (DA) Calling is an EAS feature that lets a caller:

- Contact a specific agent instead of a skill hunt group
- Queue for the agent if the agent is on a call
- Use Agent LoginID for callbacks and transfers
- Hear system wide Direct Agent delay announcement while holding
- Follow the agent's coverage path, if the call is not answered immediately.

Advantages of Direct Agent calling

DA calls have two important advantages:

- They reduce the need to transfer callers who want or need to speak with a certain agent, such as the agent spoken to on a previous call.
- They provide more accurate reporting of calls, because CMS counts DA calls as ACD calls. In this way, agents get proper credit for taking them. By comparison, calls transferred to an agent are not counted as ACD calls.

How Direct Agent calling works

Direct Agent calling works as described below:

- Callers can dial the agent's Login ID as part of a DID or from auto attendant as an extension number.
- DA calls have a special ringing sound, regardless of the agent's work state, and the current work mode button on the agent's telephone flashes.
- If the agent is on a call, he or she can use multiple call handling to decide whether to put the call on hold in order to take the DA call.
- If the agent is available, the call is delivered according to the answering and ringing options.
- If the agent is not available, or if multiple call handling is not used, call coverage or RONA routes the call to backup.
- While on DA calls, agents are unavailable for subsequent ACD calls. If the agent logs off by unplugging the headset, he or she can still answer a DA call in the queue by logging back in and becoming available. Agents who have DA calls waiting are not allowed to log off using a FAC. If the agent is in Manual In mode or pushes the After Call Work (ACW) button while on a direct-agent call, the agent goes to ACW mode.

Generally, direct-agent calls are queued and served in first-in, first-out order before other calls, including priority calls. However, if you administer a skill level for Call Handling Preference, direct-agent calls must be assigned the highest priority for them to be delivered before other ACD calls. Otherwise, calls with a higher skill level are distributed before direct-agent calls.

Note that you can use Multiple Call Handling (MCH) to allow agents to answer a direct agent call with another ACD call active.

Direct-agent calls follow the receiving agent's coverage and call forwarding paths, if these features are administered. Once a call goes to coverage or is forwarded, the call is no longer treated as a direct-agent call, and CMS is informed that the call has been forwarded.

Administering Direct Agent calling

To administer Direct Agent calling:

- On the Agent LoginID form, you enter the agent's Direct Agent Skill. It is suggested that you use the Hunt Group form to set up a skill for all DA calls. This skill will:
  - Tell the switch how to handle calls to the skill
  - Show report users how much time each agent has spent on DA calls.

  Note:

  Any agent who will receive direct agent calls should have at least one non-reserve skill assigned to the agent loginID.
- Add the skill to the agent's administered skills on this form.

Whenever an outside caller dials the agent's extension, the switch looks at the entry in that field to determine the skill for tracking call data.

On page 8 of this Feature-Related System Parameters form, you may specify:

- A Direct Agent Announcement Extension that plays an announcement to Direct Agent callers waiting in queue.
- Amount of delay, in seconds, before the announcement.

You also need to administer a Class of Restriction (COR) for DA calls. COR is covered in the next lesson. Click here to go to that topic now.

Direct Inward Dialing (DID) is administered on the Trunk Group form.

On the second page of the Hunt Group form, consider administering Multiple Call Handling On-Request for this hunt group. This feature will enable agents to see that the incoming call is a DA call and put their current call on hold to answer the DA call.

If there is no answer after a certain number of rings, you may use RONA to redirect the caller to a VDN that points to a vector. You can set up the vector to provide appropriate routing and treatment for the call.

On page 3 of the Hunt Group form, you administer messaging for the Direct Agent hunt group.

That's all. Next, you need to assign this hunt group to agents who need to receive Direct Agent calls.

Functions and examples

This section explains how EAS is implemented. Skill administration, the delivering of calls to a skill queue, and the routing of calls to an agent is discussed.

Administering skills

A skill is an attribute that is:

- Administered as a skill hunt group
- Administered to VDNs (VDN skill preference)
- Assigned to agents (agent skill)

A skill hunt group is administered for each skill. A skill hunt group is a set of agents trained to meet particular customer needs.

Generally, if the ability "Spanish speaking" is assigned to skill 127, for example, it follows that Agent skill 127 and VDN skill 127 both signify "Spanish speaking." However, note that the agent skill might be assigned a skill term that is broader than that for the corresponding VDN skill. For example, Agent skill 127 might be labeled "bilingual." The implication is that agents with skill 127 can handle calls from Spanish callers as well as from callers who speak English.

Skills for an application are shown in the following table, which presents a very abbreviated example of such a skill distribution for an automobile club.

Example of a skill table for an automobile club

Supergroup-99	
Emergency road service-bilingual-22	Route planning-bilingual-44
Emergency road service-English-11	Route planning-English-33

In the table shown above, five skills are defined. Each skill indicates knowledge or an ability on the part of the agent or a need for knowledge on the part of the caller. One or more of these skills can be attributed to the agent according to the agent's expertise with the corresponding highway services and his or her language-speaking ability. Similarly, one or more of these skills can be considered "needs" on the part of the caller.

The table shown above, is arranged in such a manner that the agents at the top level have the broadest knowledge, that is, these agents can handle emergency road service and route planning calls and can speak Spanish. The top level (skill group) here is called “Supergroup,” and it contains agents who, as a group, can take any type of call regarding the automobile club. Accordingly, this skill group serves as a “backup” skill group. As you descend through the table, each sublevel corresponds to a group of agents who have more specific skills and can therefore take more specialized calls.

Calls can be distributed to the most-idle agent by using either the Uniform Call Distribution (UCD) option or the Expert Agent Distribution (EAD) option. UCD distributes calls from the skill hunt group to the most-idle agent who has this skill assigned at any priority level. This scenario provides a more even distribution to calls and therefore keeps agents equally busy. EAD distributes calls from the skill hunt group to agents to an available agent who has the highest skill level. Skills that are assigned to an agent at higher skill levels indicate a higher level of expertise or preference by the agent than any lower skill level skills that are assigned to that agent. EAD distribution provides the caller with the best or most expert agent match.

Agents are usually given a preference for higher skill level calls. However, the system can be administered to give agents a preference for the greatest need call. The greatest need call is the highest priority oldest call waiting for any of the agent’s skills.

Multiple Call Handling on Request and Forced Multiple Call Handling make it possible for an agent to receive additional ACD calls either after putting a call on hold, or when active on another ACD call. Forced Multiple Call Handling can be used to give priority to an ACD call over an in-progress non-ACD call, or to give priority to a call from one skill over an in-progress call from a different skill. See *Administrator Guide for Avaya MultiVantage Software*, 555-233-506 for more information.

To administer skills, the relevant Hunt Group form must be completed, as shown in the following example.

Hunt Group Form

```
                                   HUNT GROUP

    Group Number:                Group Extension:          Group Type:
      Group Name:                          Skill?                 ACD?
           Queue?                         Vector?                 AAS?
   Security Code:      Night Service Destination:                 COR:
ISDN Caller Disp:                   Coverage Path:                 TN:

             Measured:                          Supervisor Extension:
  Priority On Intraflow?                      Inflow Threshold (sec):
    Controlling Adjunct:                      Adjunct Link Extension:
 Multiple Call Handling?               Acceptable Service Level (sec):
            Objective:

           Queue Length:
Calls Warning Threshold:         Calls Warning Port:          Extension:
 Time Warning Threshold:          Time Warning Port:          Extension:
     Timed ACW Interval:
                  Redirect on No Answer (rings):     Redirect to VDN:
             Forced Entry of Stroke Counts or Call Work Codes?
```

The `Skill`, `ACD`, and `Vector` fields must all contain **y**. Instructions for completing this form are included in *Administrator Guide for Avaya MultiVantage Software*, 555-233-506.

VDN skills

EAS enhances the Call Vectoring and Automatic Call Distribution features of the switch by distributing incoming calls based on:

- Specific skills that are assigned to a VDN or used in a vector, and
- Skills that are assigned to an agent

For example, a caller dials a particular number (VDN). The VDN uses a vector to queue the call to an agent with a skill that matches the VDN skill.

You can assign up to three different skills to a VDN in an order that meets your callers' needs. The first skill assigned to a VDN might be the skill that is required to best meet the needs of the customer who called the VDN. The second and third skills assigned to the VDN might represent backup skills that can also meet the callers' needs.

Skills that are administered to a VDN are commonly called VDN skill preferences. VDN skill preferences are labeled **1st**, **2nd**, and **3rd**.

Note:

While skills can be optionally assigned to VDNs, the vector controls when and to what VDN skill the call queues. The application of VDN skills is described later in this chapter.

The following table shows how skill preferences can be assigned to the five VDNs that are used for the automobile club that we discussed earlier. For each VDN, the corresponding call type and the number of the vector to which the VDN points are indicated. See Example of a skill table for an automobile club on page 354 for a description of each skill.

Example of VDN skill preferences assignments

	Skill Preferences				
Call type	**VDN**	**1st**	**2nd**	**3rd**	**Vector**
General number	6005				1
Emergency Road Service (English)	6001	11	22	99	3
Emergency Road Service (Spanish)	6002	22		99	2
Route Planning (English)	6003	33	44	99	3
Route Planning (Spanish)	6004	44	99		2

In the table shown above, note that two VDNs point to Vector 3, two VDNs point to Vector 2, and one VDN points to Vector 1. Note also that a 1st and 3rd VDN skill Preference, but no 2nd VDN skill Preference, are assigned to VDN 2222. This implies that the call to this VDN (if not already answered) will wait longer before queuing to the backup skill (Supergroup-99, in our example), provided that the vector is designed to execute accordingly.

The following table shows the skill preferences that are assigned for one specific VDN (6003) that is used for the automobile club:

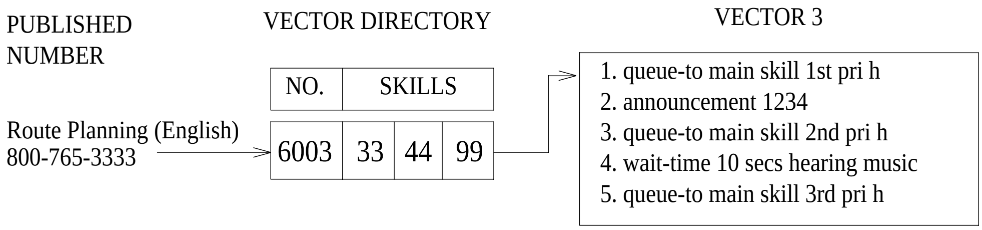
Skill preferences assignments for VDN 6003

Preference	Number	Description
1st:	33	Directed to an agent who is knowledgeable about Route Planning and speaks English
2nd:	44	Directed to an agent who is knowledgeable about Route Planning and is bilingual
3rd:	99	Directed to an agent who can field all calls

In the table shown above, the first VDN skill preference corresponds to a knowledge area that could be considered a subset of the knowledge area that is represented by the second and the third preference. Similarly, the second VDN skill Preference corresponds to a knowledge area that could be considered to be a subset of the knowledge area that is represented by the third preference. Such an approach is commonly used to assign VDN skill preferences. The result of this approach is that the longer a call waits, the larger the pool of agents that the ACD considers for handling the call.

Recall that the vector numbers for each VDN associated with the automobile club are listed in Example of VDN skill preferences assignments on page 356. VDN 6003 points to Vector 3. As such, the skill requirements that are associated with the VDN are forwarded to the vector. This process is shown in the following figure.

Example of VDN skill implementation



Assume that the English-speaking caller needs information on route planning and dials the appropriate number (800-765-3333). Network 800 features direct the call to 6003 (a VDN), the call enters the switch and is directed to VDN 6003, which points to the appropriate vector. As shown in Skill preferences assignments for VDN 6003 on page 357, VDN skill Preferences 33, 44, and 99 are administered as the 1st, 2nd, and 3rd skill preferences, respectively, for VDN 6003.

Vector processing of this application is described in Delivering the call to the skill queue on page 363.

Vector Directory Number (VDN) form

The Vector Directory Number (VDN) form shown in the following example is used to administer VDN skills.

Vector Directory Number (VDN) form, page 1

```
change vdn xxxxx                                                      page 1 of 2
VECTOR DIRECTORY NUMBER

                            Extension: 2001
                                 Name: vdn 2001
                        Vector Number: 1

                  Attendant Vectoring? n
                   Allow VDN Override? n
                                  COR: 1
                                   TN: 1
                             Measured: internal
       Acceptable Service Level (sec): 20
              Service Objective (sec):

        VDN of Origin Annc. Extension:
                            1st Skill:
                            2nd Skill:
                            3rd Skill:
```

Vector Directory Number (VDN) form, page 2

```
change vdn xxxxx                                                      page 2 of 2
VECTOR DIRECTORY NUMBER

                           Audix Name:
                 Messaging Server Name:
                    Return Destination:
                  VDN Timed ACW Interval:
                       BSR Application:
         BSR Available Agent Strategy: 1st-found
             Observe on Agent Answer?: n
```

Note:

Skills can be optionally assigned to VDNs, however, the vector controls when and to what VDN skill the call queues.

Complete instructions for completing the form are included in *Administrator Guide for Avaya MultiVantage Software*, 555-233-506.

Call Vector form

Completion of the Call Vector form is required for using vectors with EAS. The form contains three pages. However, if the vector contains 11 or fewer instructions, you need to complete only the first page of the form, as shown in the following example.

Call Vector form (Page 1 of 3)

```
change vector 20                                                Page 1 of 3
                           CALL VECTOR
 Number: 20                Name:_______________________
Multimedia? n     Attendant Vectoring? n           Lock? y
     Basic? y  EAS? y   G3V4 Enhanced? n  ANI/II-Digits? n ASAI Routing? n
 Prompting? n  LAI? n  G3V4 Adv Route? n   CINFO? n    BSR? y   Holidays? y

 01 _______________
 02 _______________
 03 _______________
 04 _______________
 05 _______________
 06 _______________
 07 _______________
 08 _______________
 09 _______________
 10 _______________
 11 _______________
```

Note:

Skills can be optionally assigned to VDNs, however, the vector controls when and to what VDN skill the call queues.

Instructions for completing the Call Vector form are provided in *Administrator Guide for Avaya MultiVantage Software*, 555-233-506, and in Creating and editing call vectors on page 29.

Agent skills

Agents are trained or hired to accommodate specific caller needs. Agent skills represent and define the ability of the agent to handle calls that require these skills. Agents are assigned skill numbers that are based on such characteristics as training or knowledge, access to systems or information, language ability, and interpersonal traits. Examples of agent skills include the following: speaks Spanish, knows about widget “X,” can handle complaint calls, has access to a database, and so forth.

You can assign up to 20 skills (with EAS-PHD) or 4 skills (without EAS-PHD). Each of these skills can be designated a skill level between 1 and 6 (EAS-PHD) or 1 and 2 (EAS), with 1 being the highest skill level, which is the highest-priority skill.

If an agent has multiple skills, a single skill group can be created for each set of skills. Agent skills are assigned to agents by completing the Agent Login ID form. See the ACD Login ID dialing for more information.

It is highly recommended that you create a separate skill hunt group for direct agent calls. Direct agent calls are queued to the skill that is administered as the Direct Agent Skill on the Agent LoginID form. If an agent is not able to log in to his or her Direct Agent Skill, Direct Agent calls are queued to the first-administered highest-level skill.

The following table shows the assignment of agent skills. See Example of VDN skill preferences assignments on page 356 for a description of the skills.

Example of agent skill assignments

Agent	Skills assigned			
Jan O’Hara	22 (L1)	44 (L2)		
Sam Lopez	99 (L1)			
Sue Carlson	22 (L1)	11 (L1)	44 (L2)	33 (L2)
Mark Davis	44 (L1)			
Amy Brown	44 (L1)	22 (L2)		

Without EAS-PHD a maximum of four agent skills may be assigned to any one agent with one of two preference levels. With EAS-PHD up to 20 skills can be assigned to each agent with one of sixteen preference levels. The skill assignments table shows that four agent skills (22, 11, 44, 33) are assigned to Sue Carlson. These assignments indicate that Sue is bilingual and can service callers who need emergency road service or information on route planning. Only one agent skill (99-Supergroup) is assigned to Sam Lopez. This means that Sam is serving only as a backup.

A L1 or L2 next to the skill number indicates whether the agent skill is assigned as a level 1 or level 2 skill. For example, Jan O'Hara has "Emergency Road Service-Bilingual" as a level one skill and "Route Planning-Bilingual" as a level two skill. This means that whenever Jan O'Hara becomes available for an ACD call, provided that the Call Handling Preference is skill-level, the ACD software first looks for English-speaking callers who are requesting information on "Emergency Road Service" from the agent. Only if there are no callers requesting "Emergency Road Service" does the ACD software look for English-speaking callers who are requesting information on "Route Planning." If the Call Handling Preference is greatest-need, Jan O'Hara receives the highest priority, oldest call waiting for either "Emergency Road Service" or "Route Planning-Bilingual" each time that she becomes available.

For any given application, EAS puts no restrictions on which agent skills can be assigned to an agent.

Note:

Agent skills are administered by completing the Agent Login ID form. This form is shown in ACD Login ID dialing on page 371. Complete instructions for completing the form are provided in *Avaya MultiVantage Call Center Software – Guide to ACD Call Center*, 555-230-716.

Preference Handling Distribution

Preference Handling Distribution enables an agent to take calls based on either skill level or greatest need.

If an agent's call-handling preference is by skill level, the agent receives the call that requires the skill for which the agent's skill level is highest.

If an agent's call-handling preference is by greatest need, the agent receives the highest-priority, oldest call waiting that requires any of the agent's skills.

It is recommended that in any skill, all agents have the same call handling preference. This ensures the most consistent distribution of calls by either greatest need or skill level.

Preference Handling Distribution Examples

The following table is an example of how calls queue with Preference Handling Distribution.

Preference Handling Distribution

Agent is assigned skills and skill levels...	These calls are in queue...
Skill 11; skill level 1	Waiting 15 seconds; priority medium
Skill 21; skill level 8	Waiting 30 seconds; priority low
Skill 31; skill level 16	Waiting 45 seconds; priority medium

Logical Agent capability

With Logical Agent and EAS, calls are routed to agents based on the login ID instead of the extension number that is assigned to the telephone. The agent's login ID must be consistent with the dial plan of the switch. When an agent logs in to an extension, the login ID overrides the extension as far as ACD tracking and characteristics, such as name and class of restriction (COR) are concerned.

When a specific login ID is called, the switch routes the call to the telephone that the agent is currently logged in to. Logical Agent allows agents to be called regardless of the telephone the agent is using. Calls to agent login IDs can be delivered as direct agent calls with the proper COR set for both the originating and the receiving login ID/facility.

Agents are not assigned to skill hunt groups with Logical Agent. Instead, an agent has specific skills that are assigned to his or her login ID. When an agent logs in, the agent is associated with the assigned skill hunt groups and tracking begins for the assigned skills.

Note:

Avaya CMS automatically measures a logical agent who is administered with at least one measured skill when the agent logs in.

Logical Agent uses a single set of work-mode buttons for all skills. This means that an agent is available or in AUX work for all skills at the same time. An agent cannot be available in some skills and in AUX work in others.

The telephone's button assignments and automatic answer options do not follow the agent because they are associated with the physical extension and not the agent login ID.

Note:

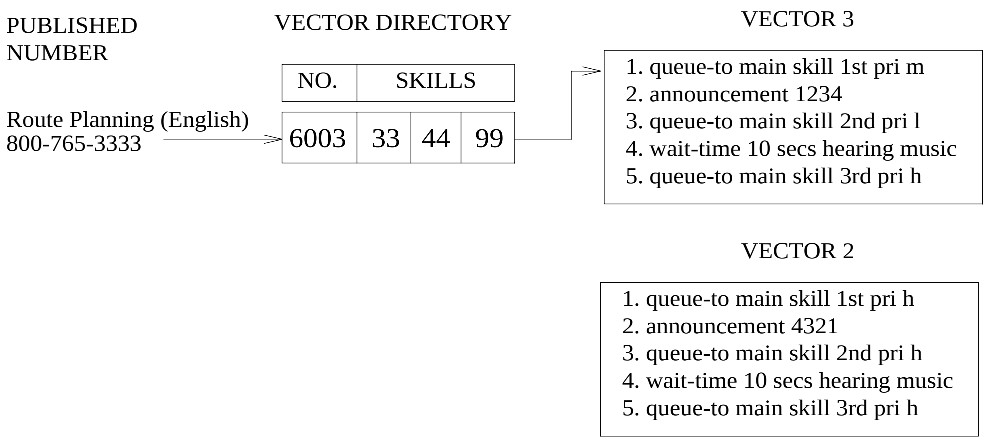
Converting to EAS may require a change to the CMS login ID if the current ID is not a valid extension number or cannot be made available in the switch dial plan. Agent login IDs are assigned names from the Dictionary-Login Identification window by way of Avaya Supervisor. Login IDs must be different from the telephone extensions.

Delivering the call to the skill queue

This example shows how a call is delivered to a skill hunt group queue by vector processing.

The skills that are assigned to a VDN define the requirements in the vector for routing calls to an ACD agent with a particular set of skills. These skills become active for an ACD call whenever a **`queue-to skill`** command is executed. The skills also become active whenever a **`check skill`** command is executed and the threshold condition is met. Once a skill is active for an ACD caller, the call cannot be delivered to an available ACD agent unless the agent also has one of the active VDN skills.

Process for delivery of a call to a skill queue



The process shown above assumes that an English-speaking caller needs information on “Route Planning” and dials the appropriate number (800-765-3333). In this case, the call enters the switch and is directed to VDN 6003, which points to Vector 3. Once vector processing starts, the **`queue-to skill`** command in step 1 queues the call to the skill hunt group that corresponds to the 1st VDN skill (33-Route Planning-English). If an agent with skill 33 is available, this agent answers the call. If such an agent is not available, the call is eventually queued to the skill hunt group that corresponds to the 2nd VDN skill (44-Route Planning-Bilingual) by the **`queue-to skill`** command in step 3. This time, if an agent with skill 44 is available, this agent answers the call. If the call is still not answered, the call is eventually queued to the skill hunt group that corresponds to the 3rd VDN skill (99-Supergroup) by the **`queue-to skill`** command in step 5.

In the process shown above, Vector 2 would be executed if a Spanish-speaking caller had called into the switch. Accordingly, the announcement that is provided in Vector 2 is in Spanish, whereas the announcement in Vector 3, which is executed in our example, is in English.

Note also that each of the **queue-to skill** commands in Vector 2 queues the call at a high priority, whereas only one of the **queue-to skill** commands in Vector 3 queues the call at this high a priority level. The strategy presented here is valuable when there is a limited number of bilingual agents because the bilingual such agents will be available more quickly to service callers who speak only Spanish.

VDN skills can also be used in **check skill, messaging skill,** and **converse-on skill** commands. Within any of these commands, a specific skill number can be used instead of a VDN skill Preference, provided that the relevant skill hunt group is correctly administered. For example, step 5 might have read **queue-to skill 99 pri h**. This concept is discussed further in Super agent pool on page 365.

Procedure using Call Prompting

The procedure that is described in the previous section can be enhanced by using Call Prompting. For example, the user can dial a general telephone number whose VDN points to a Call Prompting vector.

Staying with our automobile club example, recall that in Examples of services and corresponding DNIS digits on page 350, we define “800-765-5555” as the general telephone number for the service. Recall also that in Example of VDN skill preferences assignments on page 356 we identify **6005** as the VDN for this 800 number. Also, we indicate that VDN 6005 points to Vector 1.

The following vector shows how Vector 1 might appear.

Call Prompting vector for the automobile club

```
1.  wait-time 0 seconds hearing ringback
2.  collect 1 digits after announcement 5678
    ("For emergency road service, dial 1.
    Para asistencia con su automovil, marque el dos.
    For travel route directions, dial 3.
    Para informacion sobre rutas, marque el cuatro.")
3. route-to number 6001 with cov n if digit = 1
    (English Emergency Road Service VDN)
4. route-to number 6002 with cov n if digit = 2
    (Bilingual Emergency Road Service VDN)
5. route-to number 6003 with cov n if digit = 3
    (English Route Planning VDN)
6. route-to number 6004 with cov n if digit = 4
    (Bilingual Route Planning VDN)
7. route-to number 6002 with cov n if unconditionally
    (Bilingual Emergency Road Service VDN)
```

Once the caller dials “800-765-5555,” the call enters the switch and is directed to VDN 6005, which points to our Call Prompting vector. At this point, vector processing begins. Step 1 provides ringback if the caller has to queue for the announcement in step 2. The **`collect digits`** command in step 2 first provides an announcement that requests the caller to dial **1**, **2**, **3**, or **4**, depending upon the caller need and the caller’s language speaking ability. If the caller dials a digit that is other than one of the four specified, each of the **`route-to...if digits`** commands in steps 3 through 6 fails, and control is passed to the **`route-to...if unconditionally`** command in step 7, which unconditionally routes the call to VDN 6002. This VDN is assigned the “Bilingual Emergency Road Service” skill and points to Vector 2, which is provided in the previous section.

Now we return to the **`collect digits`** step and assume that the caller dials **4**. In this case, steps 3 through 5 fail because the required digit (**1**, **2**, or **3**, respectively) was not dialed. Thereafter, control is passed to step 6, where the **`route to...if digit`** command finds a digit match and consequently routes the call to VDN 6004. This VDN is assigned the “Bilingual Route Planning” skill and also points to Vector 2, which is provided in the previous section.

Note:

VDN Override applies to the skills that are assigned to the VDN. See VDN Override on page 52 for more information.

Super agent pool

EAS allows a skill hunt group to function as a super agent pool. A super agent pool is a backup group of one or more agents that is able to handle many if not all types of calls coming into the application. In our automobile club examples, Skill Hunt Group 99 (Supergroup) serves as a super agent pool. Also, you might recall that **99** appears as both a VDN skill and an Agent skill. However, a super agent pool can be assigned a skill hunt group number that is not assigned to a VDN skill. This can and should be done whenever the application requires four levels within the skill table distribution, as shown in the following table.

T

Modified skill table for the automobile club

Supergroup-99			
Emergency road service- bilingual-88		**Route planning-bilingual-77**	
English-66	Spanish-55	English-44	Spanish-33
Bostonian-11	Castilian-13	Bostonian-15	Castilian-17
New Yorker-12	South American-14	New Yorker-16	South American-18

Besides a new skill numbering scheme, our modified skill table has four levels instead of the three levels that are provided in Example of a skill table for an automobile club on page 354. Except for the skill numbering scheme, the top two levels (Supergroup-99 and Emergency Road Service-Bilingual-88/Route Planning-Bilingual-77) remain unchanged. However, note that the next level is reorganized into segments to indicate the ability to speak English or Spanish. Finally, note that a new level is added to denote particular types of accents or pronunciation in English and Spanish.

The following table shows how some of the skills in Modified skill table for the automobile club on page 365 are administered to one relevant VDN (VDN 1616).

Skill preferences assignments for VDN 1616

VDN 1616 - Skill preferences		
1st:	16	Knows about Route Planning, speaks English, has New York accent
2nd:	44	Knows about Route Planning, speaks English
3rd:	77	Knows about Route Planning, is bilingual

Now we are ready to consider the following vector to accommodate a super agent pool.

Modified vector to accommodate a super agent pool

```
1. queue-to skill 1st pri m
2. announcement 4555
3. queue-to skill 2nd pri l
4. wait-time 10 seconds hearing music
5. check skill 3rd pri l if calls-queued < 3
6. announcement 4666
7. check skill 99 pri l if available-agents > 0
```

Assume an English-speaking caller needs information on “Route Planning” and want to speak to an agent with a New York accent. In this case, the caller dials the appropriate number (800-765-1616, for example). Accordingly, the call enters the switch and is directed to VDN 1616, which points to the vector in the previous screen. Once vector processing starts, the **queue-to skill** command in step 1 queues the call to the skill group that corresponds to the 1st VDN skill (New Yorker-16). If an agent with skill 16 is available, this agent answers the call. If such an agent is not available, the call is eventually queued to the skill group that corresponds to the 2nd VDN skill (English-44) by the **queue to main skill** command in step 3. This time, if an agent with skill 44 is available, this agent answers the call. If the call is still not answered, the **check skill** command in step 5 attempts to queue the call according to the parameter indicated (if calls-queued < 3) to the skill group that corresponds to the 3rd VDN skill (Route Planning-Bilingual-77). If the call is queued, and if an agent with skill 77 is available, this agent answers the call. If the call is not queued, or if it is queued and an agent with skill 77 is not available, the **check skill** command in step 7 is executed.

Before we discuss the execution of step 7, note that a specific skill hunt group number (99) and not a VDN skill Preference designation (1st, 2nd, or 3rd) is included within the **check skill** command. Since the skill table for the application involves four levels of skills, and since there can be no more than three VDN skills, the specific skill group number (99) for the super agent pool must be included within the queuing command to allow caller access to the pool. Whereas a VDN skill is always represented in a vector by the term 1st, 2nd, or 3rd, a super agent pool is always represented by a whole number according to the parameters of the relevant switch. See Appendix A: Call Vectoring commands on page 387 for the queueing commands.

Returning to the vector execution, the **check skill** command in step 7 attempts to queue the call according to the parameter that is indicated (if available-agents > 0) to the super agent pool (Supergroup-99). If the call is queued, and if an agent in the super agent pool is available, this agent answers the call.

Note:

If the call has already queued to all three VDN skill hunt group preferences, it does not queue to the specific skill hunt group. This reflects the restriction that a call can only queue to a maximum of three splits or skills. The best approach is to test the splits/skills first to determine where to queue the call. Also, see Expected Wait Time (EWT) on page 126.

Routing the call to an agent

With EAS optioned, an agent becomes associated at login with one or more skill hunt groups. A single set of work mode buttons applies to all the skills that are assigned to a logged-in agent. For example, if the agent selects **Aux Work**, the agent is in Aux Work for all the skills associated with the agent. Therefore, logged-in agents need only a single set of work-mode buttons for all relevant skill hunt groups.

Calls can be routed to the agent from a skill hunt group by dialing an Agent Login ID or by dialing an agent telephone extension directly. The following sections discuss these procedures.

Delivery from a skill hunt group

An incoming call is matched to an agent who has at least one of the three VDN skills that are required to handle the call. This matching is done by ACD queuing and the **queue-to skill, check skill, messaging skill,** or **converse-on skill** commands in the vector. If more than one agent is available for a call, the call is delivered according to whether EAD or UCD is administered for the skill hunt group.

For any one login session, an agent can have a maximum of four skills, or a maximum of twenty skills with EAS-PHD. Each agent skill is administered with a skill level.

Remember that when the Call Handling Preference is administered as greatest need, the agent receives the highest priority oldest call waiting for any of the agent's skills. If the Call Handling Preference is skill-level, the ACD software distributes the call that is waiting for the agent's highest skill-level skills whenever the agent becomes available. If no calls are waiting for the highest skills, the queued calls for the next highest skills are distributed to the agent, and so on. The following scenario describes call distribution when the Call Handling Preference is skill level.

Once an agent becomes available, he or she receives a waiting call in the following order:

1. Oldest Direct Agent call waiting for the agent if the Direct Agent Skill is administered at the agent's highest skill level
2. Oldest call waiting at the highest priority for the highest skill-level skill
3. Oldest call waiting at the next highest skill-level skill, and so on.

For example, assume that Jill is the only agent with skills 22 (L1), 13 (L1), 23 (L1) and 47 (L2). Also assume that, while Jill is in AUX work mode, five calls are queued, as shown in The following table, which also shows the skill level and priority level that are associated with each call:

Example of skill call queue sequence

Call	Time in queue	Skill number	Priority level
A	8:00	13	Medium
B	8:01	47	Top
C	8:02	23	Direct Agent
D	8:03	22	Top
E	8:04	22	Medium

Given this scenario, the next table indicates and explains the order in which Jill handles the five calls.

Example of skill call distribution for a single agent

Call handled	Reason
C	Only Direct Agent call queued at highest level skill.
D	Oldest call waiting at the highest priority for highest skill-level skills (Call B has the same priority level (Top), but it is assigned a lower skill level (47). Also, Call E has the same skill (22), but it has a lower priority level (Medium) and has not been waiting as long as Call D).

Example of skill call distribution for a single agent (continued)

Call handled	Reason
A	Oldest call waiting at the highest priority level for highest skill-level skills (Call E also has a primary skill (22) and the same priority level as Call A, but Call A has been waiting four minutes longer than Call E).
E	Only remaining call with the highest skill level (22) (Call B has a lower skill level (47)).
B	Last remaining call, and the only one that has the lower skill level (47).

If no calls are waiting when an agent becomes available, the agent is placed into the agent queue according to the call distribution method that is in effect. For UCD, the agent is placed at the bottom of the most-idle agent queue. For EAD, the agent is placed at the bottom of the agents with the same skill level.

The following table shows a call scenario that is valid for either UCD or EAD.

Example of UCD/EAD call scenario

Time	Event	Skills
9:00	Jill logs in	22(L1), 13(L1), 47(L2)
9:01	Jill available	22(L1), 13(L1), 47(L2)
9:02	Jack logs in	22(L1), 47(L1)
9:03	Jack available	22(L1), 47(L1)
9:04	Call A arrives	47
9:05	Call A drops	47
9:06	Call B arrives	13
9:07	Call B drops	13
9:08	Call C arrives	22

Given the scenario presented above, the following table shows how Calls A, B, and C are distributed by UCD and EAD:

Example of call distribution by UCD and EAD

Time	UCD or EAD?	Result	Reason
9:04	UCD	Jill receives Call A.	Jill is the most idle agent for skill 47.
	EAD	Jack receives Call A.	Jack is the “more expert” agent because he has skill 47 as a level 1 skill whereas Jill has skill 47 as a level 2 skill.
9:06	UCD	Jill receives Call B.	Jill is the only agent who is logged in to skill 13.
	EAD	Jill receives Call B.	Jill is the only agent with skill 13.
9:08	UCD	Jill receives Call C.	Jill is the most idle agent for skill 22. She receives Call C even if she handled Call A.
	EAD	Jill receives Call C.	Both Jill and Jack have skill 22 as a level 1 skill, but Jill has been logged in 2 minutes longer than Jack; that is, she is the most idle agent.

ACD Login ID dialing

The ACD Login IDs used in EAS are extension numbers that are included in a station numbering plan but not administered as stations. These IDs are administered by using the Agent Login ID form, as shown in the following example. If EAS-PHD is not optioned, you can only administer four skills.

Agent Login ID form

```
add agent-loginID 9011                                            Page  1 of  1
                             AGENT LOGINID

                Login ID: 9011_                                    AAS? _
                    Name: _________________________              AUDIX? _
                      TN: 1__                            LWC Reception: spe
                     COR: 1                   AUDIX Name for Messaging: _______
           Coverage Path: ____   Messaging Server Name for Messaging: _______
           Security Code: ____                LoginID for ISDN Display? n
      Direct Agent Skill: ___                                Password: _________
Call Handling Preference: skill-level        Password (enter again): _________
       Service Objective? _                               Auto Answer: ______

   SN  RL SL   PA      SN  RL SL   PA      SN  RL SL   PA      SN  RL SL   PA
1: __  _  __  ___  6: __  _  __  ___  11: __  _  __  ___  16: __  _  __  ___
2: __  _  __  ___  7: __  _  __  ___  12: __  _  __  ___  17: __  _  __  ___
3: __  _  __  ___  8: __  _  __  ___  13: __  _  __  ___  18: __  _  __  ___
4: __  _  __  ___  9: __  _  __  ___  14: __  _  __  ___  19: __  _  __  ___
5. __  _  __  ___ 10: __  _  __  ___  15: __  _  __  ___  20: __  _  __  ___

      WARNING:  Agent must log in again before skill changes take effect
```

With EAS, an Agent's ACD Login ID is associated with a specific telephone only when the agent actually logs in at that telephone. When the agent logs off, the association of the agent's ACD Login ID with a specific telephone is removed. If an agent does not answer a call, or if the agent is logged out, the call goes to the busy points on the coverage path.

When the agent logs in, the telephone display indicates the agent's skill assignments.

The agent logs in by doing the following:

- Going off-hook or selecting a line appearance
- Upon hearing the dial tone, entering the login Feature Access Code (FAC) or selecting the Login Abbreviated Dialing (AD) button
- Upon hearing the dial tone, entering the 1-digit to 5-digit Login ID

  Note:

  If someone is already logged in at that telephone, the agent hears an intercept tone.

- Upon hearing the dial tone, entering (optionally) the 0-digit to 9-digit password.

  Note:

  If the agent is using a DCP telephone (such as a Callmaster), then the password digits are not shown unless an abbreviated dial button is used. BRI telephones show the password digits.

Once the login is accepted, confirmation tone is given. Also, the skills that are assigned are displayed for 5 seconds on the telephone display. If more skills are assigned than can be displayed, a plus sign (+) appears at the end of the display. If a skill is administered but the agent was not logged in to the skill, the skill number is displayed with a star (*). The previous login sequence allows an ACD call to be directed to a specific agent and to have that call tracked and treated as an ACD call.

When an EAS agent logs in to a station with the station administered for audible message waiting, the agent receives an Audible Message Waiting tone only when calls are waiting for the agent login ID extension. When the agent logs out, Audible Message Waiting tone then applies again to messages that are waiting for the physical extension. This field has no impact on whether an agent hears the EAS Login-ID Message Waiting tone during the login process.

The message waiting lamp by default tracks the status of messages that are waiting for the logged-in EAS agent LoginID rather than messages for the physical telephone. The operation of the Message Waiting Lamp can be changed so that it tracks the status of messages that are waiting for the physical telephone where the agent is logged in. For more information, see the Feature-Related System-Parameters form in *Administrator Guide for Avaya MultiVantage Software*, 555-233-506.

Other agent login capabilities

In addition to skill assignments, the following capabilities are associated with agents' login IDs.

Call routing – A call to the Login ID reaches the agent independent of the telephone that the agent is currently using. In other words, such a call is sent to the telephone at which the agent is currently logged in.

If the proper Class of Restrictions (COR) is set, callers can initiate a Direct Agent call either by dialing the Login ID extension directly or by calling a VDN that points to a vector that contains first a prompt for the Login ID and then a **`route-to digits`** command. This allows external callbacks by way of Direct Inward Dialing (DID) or an 800 number. Both the receiving agent's Login ID COR and the originator's (caller's) COR must have Direct Agent Calling set to **y**. The caller's COR is for the following:

- Telephone extension (for internal calls or transfers)
- Trunk group (for DID calls)
- VDN (for prompted calls)

If the call covers or is forwarded, the COR of the originator (or VDN) and the final agent is used. All feature functionality for ACD calls, except Queue Status indications, is available for Direct Agent Calls.

Internal and external users can originate Direct Agent calls by dialing the agent's login ID. Also, Direct Agent calling can be used to transfer ACD calls from one agent to another agent.

If an agent who is receiving the Direct Agent Call is staffed but unavailable, the call waits in front of the skill calls in the skill that is administered as the agent's Direct Agent Skill until either the call is answered or a coverage timeout occurs. Also, the caller hears an optional direct-agent announcement that is followed by music or silence. There is one Direct Agent announcement per system. The agent, on the other hand, receives a ring-ping, and the current work mode button flashes. If the agent is available, the call is delivered to the agent according to the answering and ringing options. Calls are answered and handled in the same manner as ACD calls. See the Feature-Related System-Parameters form in *Administrator Guide for Avaya MultiVantage Software*, 555-233-506 for more information.

Login ID name on the telephone display – A call to a logged-in EAS Login ID by default displays the name associated with the Login ID and not the name that is associated with the telephone. This is also true on the receiving party's display for a call that is made from a telephone with an agent logged in. However, the user can display the name of the physical telephone where the EAS agent is logged in. The user must be active on a call with the agent, and must have a telephone with an alphanumeric display and an inspect button. When the inspect button is pressed during a call to or from the EAS agent, the physical telephone name of the agent is displayed.

Coverage path – Call coverage can occur whether or not the agent is logged in. If the agent is not logged in, the busy criteria is met and the call follows the points on the coverage path. If the agent is logged in but fails to answer, the don't answer criteria is met and the call follows the points on the coverage path. A call to the Login ID goes to the coverage path that is assigned to the Login ID rather than to the coverage path that is assigned to the telephone extension.

Agent restrictions – A call to the Login ID or from the agent uses the restrictions that are associated with the agent and not the telephone.

Telephones are fully functional if an agent is not logged in. The restrictions, coverage, and name revert to the telephone administration when the agent logs out.

If a number of users are sharing one telephone (due to job sharing or shifts, for example), a unique Login ID extension is assigned to each user. Therefore, whenever a user is logged out, any calls to that user (login ID) are sent to his or her coverage path. As a result, Login IDs can be used to reach people independent of where they happen to be. Such people include those who use more than one phone because they have more than one office or (in the case of security guards, for example) sit at more than one desk.

Because AAS/AUDIX ports are not mobile, these ports are administered to Agent Login IDs. Whenever the `AAS` or `AUDIX` field is set to **y**, a field that requests the port number is brought up, and the `password` field disappears.

Interactions that involve EAS

This section discusses the feature and adjunct interactions that involve EAS.

Feature interactions

This section discusses the feature interactions that involve EAS. Unless otherwise specified, the feature interactions for skill hunt groups are the same as for vector-controlled splits.

Abbreviated dialing – Abbreviated dialing can be used to log in or log out EAS agents. Abbreviated dialing lists or buttons can be administered only for stations.

Administration without hardware – Although EAS Login IDs are extensions without hardware, they are not a part of the Administration without Hardware feature.

Agents in multiple splits feature – With EAS, the Agents in Multiple Splits feature is called Agents in Multiple Skills. This feature allows an EAS agent to be logged in to multiple skills.

Agent work modes – With EAS optioned, an agent can be in only a single work mode for all skills at any one time. For example, an agent cannot be in AUX work mode in one skill hunt group and also available in another skill hunt group. Also, if the After Call Work (ACW) mode button is selected, the agent is placed into ACW for the first skill that is administered and logged in to.

Assist – This feature is used for skill hunt groups (that is, there is one supervisor per skill hunt group). A telephone can be administered with one or more Assist buttons for each skill that agents who are using the telephone might have. An Assist button can also be administered with no associated skill. In this case, the supervisor for the skill that the agent is currently active on is called. If the agent is not active on any skill, the supervisor for the agent’s first skill is called.

Any assist button that is selected is tracked as an assist for the current call, regardless of any skill that is assigned to the button. The administered association of an Assist button with a particular skill and assigned supervisor is not affected when an EAS agent logs in to that station.

Audible message waiting – If messages are waiting for an EAS agent login-ID extension, an agent hears a special 5-burst EAS Login-ID Message Waiting tone (instead of confirmation tone) after successfully logging in. This does not require Audible Message Waiting to be assigned to the telephone or the system.

If Audible Message Waiting is optioned for the system and assigned to an agent's telephone, and messages are waiting for the agent login ID extension, the agent hears the Audible Message Waiting tone whenever the agent goes off-hook, or selects a line appearance and hears dial tone. Messages that are waiting for the physical extension do not cause an Audible Message Waiting tone when an EAS agent is logged in.

Auto-Available Skills – If a skill hunt group is administered as an Auto-Available Skill (AAS), the EAS Login IDs that are assigned to this skill must also be administered as Auto-Available. When the switch reinitializes, these Login IDs are automatically logged in with the AUTO-IN work mode. If any switch features attempt to change the work mode to anything except AUTO-IN, this attempt is denied. Agents cannot have both Auto-Available and Non-Auto-Available Skills. This feature is not intended for human agents.

Automatic answering with zip tone – This feature can be administered only for a physical extension. The feature is not associated with a LoginID.

BCMS – The BCMS user interface remains the same when EAS is optioned. The only change is that the labeling of the headings is changed from split to skill. When EAS is enabled, BCMS agent reports are based on the Agent Login IDs.

BCMS tracks Direct Agent calls as skill calls. Direct Agent calls affect ACD talk time, ACW time, and Average Speed of Answer. Whenever Direct Agent calls are waiting, BCMS displays an asterisk (*) immediately after the CALLS WAITING column.

Best Service Routing (BSR) – EAS VDN skills (1st, 2nd, 3rd) can be used in **`consider split/skill`** commands. EAS skills levels are used for the EAD-MIA and EAD-LOA BSR Available Agent Strategies.

Bridging – ACD calls do not alert on bridged appearances. However, bridged users can activate features on behalf of agents. Features that can be activated include log in, log out, change work modes, and assist.

Call coverage – Call coverage can occur whether or not the agent is logged in. If the agent is not logged in, the busy criteria is met and the call follows the points on the coverage path. If the agent is logged in but fails to answer, the don't answer criteria is met and the call follows the points on the coverage path. A call to the Login ID goes to the coverage path that is assigned to the Login ID rather than to the coverage path that is assigned to the telephone extension.

Call Detail Recording (CDR) – For skill calls, the `called party` field can optionally be the Agent Login ID.

Call forwarding – Since they are vector-controlled, skill hunt groups cannot be call forwarded. EAS agent Login IDs cannot be forwarded, but the physical extension where the EAS agent is logged in can be forwarded. If another station with console permissions tries to forward an EAS Login ID, an intercept tone is given.

Call park – To retrieve a parked call by a Feature Access Code (FAC), the agent dials the Answer-Back FAC and the extension where the call is parked. If the person who is unparking the call dials the Answer-Back FAC and the physical extension of the station where the call is parked, he or she is connected to the parked call.

In some cases, the person who is unparking the call may also be able to dial the Answer-Back FAC and the logical agent extension of the agent who parked the call. This operation is possible if the Class of Restriction (COR) of both the agent parking the call and the telephone or agent who is unparking the call have a COR with the Direct Agent Calling flag set to **y**. If the telephone that is unparking the call is not a logged-in agent, the telephone must have a COR with Direct Agent Calling set to **y**. If the station that is unparking the call is a logged in agent, then the COR of the logical agent extension must have Direct Agent Calling set to **y**.

Call pickup – Skill hunt group extensions and EAS Login ID extensions cannot be members of a Call Pickup group.

Class of Restriction – Skill hunt groups do have a Class of Restriction (COR). The COR is used if the skill hunt group extension is called directly. The COR for an EAS agent Login ID overrides the physical extension's COR of the telephone that an agent logged in to.

Class of Service – EAS agents do not have a COS associated with their Login ID. Instead, the COS is associated with the physical extension. Therefore, the COS of the telephone is not affected when an EAS agent logs in to that telephone.

Dial plan – Agent Login IDs are part of the dial plan, and they reduce the total number of stations.

Direct Agent Calling (DAC) – If a called EAS Agent Login ID and the call originator (extension, trunk, or VDN) both have a COR that allows Direct Agent calls, the call to the Login ID is treated as a Direct Agent call. A call to the telephone extension where an EAS agent is logged in, or a call to an EAS Agent Login ID where either the originator's or the Login ID's COR does not allow Direct Agent calls, is treated as a personal (non-ACD) call.

Telephone displays – When an EAS agent is logged in, the display for originators who call the Login ID shows the Login ID and agent name as they are administered on the Agent Login ID form. Calls that are originated by the agent show the Agent Login ID and agent name at the receiving telephone's display. However, the user can display the name of the physical telephone where the EAS agent is logged in. The user must be active on a call with the agent, and must have a telephone with an alphanumeric display and an inspect button. When the inspect button is pressed during a call to or from the EAS agent, the physical telephone name of the agent is displayed. Calls to the physical extension show the physical extension's number and name on the originator's display.

Look-Ahead Interflow – Skills are not sent to another system when a call interflows using Look-Ahead Interflow (LAI). If skills have the same meaning on both ACDs, a LAI command to a VDN with the same skills assigned can provide a mapping of the skills.

Multiple Split Queuing – When EAS is enabled, the Multiple Split Queuing feature is called Multiple Skill Queuing, which has the same functionality. With Multiple Split/Skill Queuing, a call can queue to a maximum of 3 splits/skills.

OCM/EAS – If EAS is enabled on the switch, the Outbound Call Management (OCM)/Expert Agent Selection (EAS) feature is required for a CallVisor ASAI adjunct application to launch predictive Outbound Call Management (OCM) calls. Predictive Calling is an OCM feature that is often used in applications, such as sales or “cold calling,” where it does not matter which agent is accessed by a caller and for which it is important to keep the agents utilized fully.

While OCM predictive calling is an outbound call management application, the EAS environment provides a number of desirable features for inbound call handling. The OCM/EAS feature allows the customer to enable both types of call handling on the switch. From a technical standpoint, if EAS is enabled, the feature is needed for the following reasons:

- All skill hunt groups are vector controlled. However, to launch a predictive OCM call in a traditional ACD environment, the ACD split cannot be vector-controlled.
- The traditional ACD environment and EAS cannot be enabled on the switch at the same time.

The OCM/EAS feature extends the ASAI features to include launching predictive OCM calls from a VDN extension. Previously, ASAI hosts could launch predictive calls only from ACD split extensions. A limited number of Call Vectoring commands are supported in the VDNs that are used to launch or process OCM predictive calls. These commands are listed in the following section.

Commands for OCM predictive calls

Vectors that are intended for processing predictive calls must be designed in such a manner that the vectors are limited to the supported steps.

The following table lists vector commands available for processing OCM Predictive Calls and provides a brief comment for each command.

Commands for OCM predictive calls

Command	**Comment**
queue-to skill (single occurrence)	This command queues the call for handling by an agent in the skill pool.
announcement	This command plays an announcement if there are no agents available and if the queue (if any) is full.

Commands for OCM predictive calls

Command	**Comment**
stop	This command ends vector processing. The command also disconnects any call that is not queued.
adjunct routing	EAS supports adjunct routing to any of the following: skill extension, direct agent call, announcement, or local extension. The command does not involve routing to an external number.
wait-time	**NOTE:** This command is used with the **`adjunct routing`** step to determine how long the switch waits for an adjunct route before continuing with vector processing.

Queue Status Indications – Physical extensions can be administered with Queue Status Indicator buttons and lamps for skill hunt groups that operate in the same manner as split Queue Status Indicators for traditional ACD splits. As long as enough buttons are available, Queue Status Indicators can be administered for all skills that are needed by agents who are using that physical extension. Also, any waiting Direct Agent calls are not reflected in the queue status indicators.

Reason codes – With Reason Codes, an EAS agent can enter a numeric code that identifies his or her reason for entering AUX work mode, or for logging out.

Service Observing – Service Observing is activated in the EAS environment by dialing either the physical extension of the telephone where an EAS agent is logged in, or the EAS agent Login ID.

Remote Service Observing – Remote access to the Service Observing (SO) FACs can be provided by the Remote Access feature or through Service Observing vectors. See Service Observing routing on page 120 and Creating Service Observing vectors on page 194 for additional information.

VDN Override – If VDN Override is set to **y** (yes) on the previous VDN, the VDN skills of the current VDN are used. If VDN Override is set to n (no) on the previous VDN, the VDN skills of the previous VDN are used.

VuStats – VuStats can display information for all 20 agent skills.

Work mode buttons – Only a single set of agent work mode buttons is needed. If multiple buttons are assigned, all lamps for that work mode, for example, manual-in, light whenever any one button is pushed.

Adjunct interactions

This sections discusses the adjunct interactions that involve EAS.

ASAI

ASAI support for EAS may be organized into the following categories: call control, feature requests, value queries, event notification, and adjunct-controlled skills. This section provides a high-level overview of the behavior of ASAI in the EAS environment.

Call control

Call control capabilities work exactly the same in the EAS environment as in the traditional ACD environment except for the following:

- User-classified third party make calls (calls classified by originator) may originate from an EAS Login ID and terminate to a Login ID. User-classified calls that terminate to a Login ID are given the same Direct Agent treatment that is provided for such calls that are dialed from a station extension.
- Switch-classified third party make calls, which are classified by a call classifier board and delivered (when answered) to the originating hunt group, may originate from or terminate to EAS Login IDs.
- Direct Agent third-party make calls, which are ACD calls that are terminated to a selected member of an ACD skill group, may be requested by including a Direct Agent option, an agent’s physical extension and a skill group extension (compatibility mode), or by requesting a user-classified third-party make call with a Login ID destination. The primary differences between the two methods of requesting Direct Agent calls are that the compatibility mode allows the adjunct to specify the skill hunt group to which a given Direct Agent call is queued and that the noncompatibility mode allows the adjunct to direct the call to a Login ID, regardless of which station an agent is logged in to. Direct Agent third-party make calls may not originate from an EAS Login ID.
- Supervisor assist third party make calls, which are supervisor assist calls that are originated by a selected member of an ACD split, may originate from an EAS Login ID, and they may terminate to an EAS Login ID. Unlike dialed Direct Agent calls, supervisor assist calls that are terminated to a Login ID behave as though they have been previously directed to the requested Login ID’s physical extension. For example, they do not cover if the requested agent is not logged in and if the originator’s display shows the agent’s physical extension and not the agent’s Login ID.
- Extension (Domain) control may not be requested for an EAS Login ID, but it may be requested on behalf of a Logical Agent’s physical extension. Auto-dial calls, which are calls that are initiated by an extension-controlled station, may be terminated to an EAS Login ID, in which case the call is given Direct Agent treatment.

- Adjunct routing calls, which are vector calls that are routed by an ASAI adjunct by the `adjunct routing` Call Vectoring command, are similar to third party make calls. Such calls may include a Direct Agent option, an ACD agent's physical extension, and a skill extension. If this is true, these calls are given compatibility mode Direct Agent treatment and may be terminated to an EAS Login ID (in which case they behave like dialed Direct Agent calls.
- If EAS is optioned, ASAI launches OCM switch-classified or predictive calls from a VDN extension by the OCM/EAS feature. To launch a predictive call in a traditional ACD environment, an adjunct OCM application sends an ASAI request to the switch with an ACD split number as the originating number. The application also sends flags that identify the call as a switch-classified call. In the traditional ACD environment, the ACD split cannot be vector-controlled.

Feature requests

In the EAS environment, agent login, logout and change work-mode requests are fully supported. Agent login requests must contain an EAS Agent Login ID and optional password (delimited by '#') in the login request's user code IE. Agent logout requests and change work-mode requests may contain the desired agent's physical extension or Login ID. Call Forwarding and Send all Calls feature requests are denied for EAS Login IDs but may be requested for EAS physical extensions where an EAS agent is logged in.

Multiple monitors

Multiple Monitors provides the ability for up to three ASAI applications to monitor the same ACD Split or VDN domain.

This is helpful in environments were OCM is primary and it can also be used to add an OCM application to launch calls at off-peak times without disrupting the primary application in any way. Multiple Monitors can also be used to monitor an ACD split over 2 links in call environments where ASAI link failure recovery is important.

Value queries

Value queries function identically in the EAS and traditional environments, except that the Extension Type/Class Information Query returns a new indication that a requested extension is an EAS Login ID along with an indication of whether the Login ID is currently logged in and where, in other words, at which physical extension.

Event notification

Because all skill hunt groups are vector controlled, event notification may not be requested on the basis of a skill hunt group extension. Event notification may, however, be requested on the basis of a controlling VDN extension. Generally, all event reports that involve EAS agents contain the agent's physical extension rather than the agent's Login ID.

Adjunct-controlled skills

Agents with adjunct-controlled skills are considered to be adjunct-controlled agents. Adjunct-controlled agents exhibit the same behavior as agents within adjunct-controlled splits in the traditional ACD environment. The following list provides more details:

- Stations are locked for all logged-in adjunct-controlled agents. The only action an agent can take from the station is to go on hook (or unplug the headset) from an auto-answer station, which causes the agent to be logged out.
- Stations are unlocked whenever the controlling adjunct's ASAI link stops functioning. Stations are locked again when the adjunct's link is reestablished.
- The adjunct controls all skill and agent activities such as login, logout, and change work-mode (with the exception of agent logout using the telephone hook).
- Only adjunct-controlled calls can terminate to the extension of an adjunct-controlled agent.
- Only adjunct-controlled calls can terminate to an adjunct-controlled skill hunt group extension.
- Adjunct-controlled EAS Agents can be administered with only one skill. Accordingly, EAS agents may not mix adjunct-controlled and non-adjunct-controlled skills.

AUDIX

Calls to the EAS Agent Login ID can cover to AUDIX. Each agent must enter his or her Agent Login ID when calling AUDIX to obtain messages.

AUDIX agents are assigned to EAS agent extensions. These Login IDs are used for CMS and BCMS tracking if the associated AUDIX skill hunt group is externally measured. The **aut-msg-wt** button or **message waiting** light can be used to indicate that the Login ID has a message.

An agent cannot have both AUDIX and non-AUDIX skills.

CMS

The following is true for Avaya CMS Agent Tables:

- Separate Direct Agent database items starting with “DA_” are tracked.
- Standard reports combine statistics for Direct Agent calls and skill calls. However, reports can be customized to separate these statistical groupings.

The following is true for the CMS Skill Tables:

- Skill queues can be monitored for Direct Agent calls on the “Queue/Agent Summary” report.
- Direct Agent calls are not tracked.
- Agent time while on a Direct Agent call is tracked as “other” time.
- Non-ACD calls while in Direct Agent ACW are tracked.

The following is true for the CMS VDN/Vector Tables:

- Direct Agent calls and skill calls are combined as ACD calls.

Speech-processing adjuncts

Speech-processing adjuncts that have a line interface to the switch are able to initiate Direct Agent calls by dialing the Login ID for an agent.

Other forms that support EAS Agent LoginID

The following table lists switch administration forms that can have an EAS Agent loginID administered on them.

EAS loginID table

Feature	Accepts loginID?
Abbreviated Dialing Buttons	
7103A	Yes
Enhanced	Yes
Group	Yes
Personal	Yes
System	Yes
Agent-LoginID	
Port Extension	No
Announcements	No
Buttons	
abrdg_app	No
aut-msg-wt	Yes
brdg_app	No
busy-ind	Yes
data_ext	No
man_msg_wt	No
q-calls	No
q-time	No
signal	No

EAS loginID table (continued)

Feature	Accepts loginID?
Call Processing	
Auto-Callback	No
Call Forward from Agent Login ID	No
Call Forward to Agent Login ID	Yes
Call Park	Yes
Hundreds group	No
LWC Retriever gets lagt msgs	Yes
Service observ Agent Login ID	Yes
CDR Parameters	
Primary Extension	No
Secondary Extension	No
Code-Calling	Yes
Communication Link Form	
Communication Link Digits	No
Console Parameters	
CAS-backup ext	No
IAS Att Access Code	No
Coverage Groups	
Answer Group Member	No
Path	Yes
Measured Principals	
Coverage Measurement	No
Feature-Related Parameters	
ACA-referral dest.	No
ACA - long holding	No
ACA - short holding	No
Controlled out restriction	No

EAS loginID table (continued)

Feature	Accepts loginID?
Controlled Terminal	No
Controlled Stn-to-Stn	No
DAA Extension	No
DID/Tie/ISDN announcement	No
Emergency Access Redirection	No
CDR output extension	No
SVN referral destination (announcement)	Yes
System LWC retriever	No
System Printer	No
Hospitality Parameters	
Journal Printer	No
LWC wakeup	No
PMS ext	No
PMS log	No
Routing on Voice Synthesis	No
Hunt Group Form	
Announcement extension	No
ASAI link	No
AUDIX extension	No
Calls Warning extension	No
Member	No
Night Service	No
Supervisor	Yes
Time Warning extension	No
Intercom Group Member	No
Intra-switch CDR	Yes
Listed Directory Number	

EAS loginID table (continued)

Feature	Accepts loginID?
Member	No
Night Destination	Yes
Malicious Call Trace	
MCT Member	No
Permanent Switched Calls	No
Personal CO Line	No
Pickup Group Member	No
Remote Access Extension	No
Term Extension Group Member	No
Trunk Group	
Night Service	Yes
Incoming Destination	Yes
Member Night Service	Yes
Vector Administration	
adjunct extension	No
announcement	No
messaging	Yes
route-to	Yes

Upgrading to the EAS environment

For information on converting a call center to EAS, refer to Appendix M: Converting a Call Center to EAS on page 617.

Appendix A: Call Vectoring commands

This appendix provides information about the commands used in Call Vectoring, including:

- A table that lists the MultiVantage options that are required to enable various vector commands
- A table that contains a brief description of each command's function and also the appendix page where the command can be referenced
- A job aid tables that graphically illustrate how to use the Call Vectoring commands
- A manual page directory that details the purpose and function of the Call Vectoring commands and also any relevant interactions involving the commands

About MultiVantage Call Center packages

Some Call Vectoring commands require various software to be enabled. The features required to enable vector commands are included in the following Avaya MultiVantage Call Center Solution packages:

- Avaya Call Center Deluxe
- Avaya Call Center Elite

Most of the features required to fully enable vector commands are included in the Avaya Call Center Deluxe package. To use skill options associated with some vector commands, the Avaya Expert Agent Selection (EAS) feature must be enabled. The EAS feature is included in the Avaya Call Center Elite package. When a vector command requires the EAS featre, the requirement is noted.

In addition, other vector commands require Virtual Routing, which activates Look-Ahead Interflow. Other commands are available with non-call center right-to-use (RTU) offerings, such as AutoAttendant, which activates Prompting.

MultiVantage options required to enable vector commands

The following table lists the MultiVantage options that are required to enable various vector commands, options, and parameters.

Vector commands / required options summary

Command	Basic	Prompting	Attendant	Other Options Required
`adjunct routing`	x			ASAI
`announcement`	x	x		
`busy`	x			
`check best`	x			ACD; G3V4 Advanced Routing; Best Service Routing
`check split/skill if <condition>`	x			ACD
`check split/skill if rolling-asa`	x			ACD; G3V4 Enhanced; G3V4 Advanced Routing
`check split/skill if expected-wait`	x			ACD; G3V4 Enhanced; G3V4 Advanced Routing
`check best if expected-wait`	x			ACD; G3V4 Enhanced; G3V4 Advanced Routing; BSR
`check split/skill if oldest-call-wait pri`	x			ACD; G3V4 Enhanced
`check split/skill/best if wait-improved`	x			ACD; G3V4 Advanced Routing; Best Service Routing
`collect digits`		x		
`collect ced/cdpd digits`		x		Vectoring (CINFO)

Vector commands / required options summary (continued)

Command	Basic	Prompting	Attendant	Other Options Required
`consider location`	x			ACD; G3V4 Advanced Routing; Best Service Routing; Look-Ahead Interflow[1]
`consider split/skill`	x			ACD; G3V4 Advanced Routing; Best Service Routing
`converse-on split/skill`	x			
`converse-on split/skill passing wait`	x			ACD; G3V4 Enhanced; G3V4 Advanced Routing
`disconnect`	x		x	
`disconnect after announcement <extension>`	x		x	
`goto step/vector if unconditionally`	x	x		
`goto step/vector if <condition> in split/skill`	x			ACD
`goto step/vector if digits`		x		
`goto step/vector if time-of-day`	x			
`goto step/vector if oldest-call-wait pri`	x			ACD; G3V4 Enhanced
`goto step/vector if rolling-asa`	x			ACD; G3V4 Enhanced; G3V4 Advanced Routing
`goto step/vector if expected-wait`	x			ACD; G3V4 Enhanced; G3V4 Advanced Routing
`goto step/vector if expected-wait for best`	x			ACD; G3V4 Enhanced; G3V4 Advanced Routing; Best Service Routing
`goto step/vector if counted-calls`	x			G3V4 Enhanced; G3V4 Advanced Routing
`goto step/vector if ani`	x			G3V4 Enhanced; G3V4 ANI/II-Digits Routing

Vector commands / required options summary (continued)

Command	Basic	Prompting	Attendant	Other Options Required
goto step/vector if ii-digits	x			G3V4 Enhanced; G3V4 ANI/II-Digits Routing
goto step/vector if wait-improved	x			ACD; G3V4 Advanced Routing; BSR
goto step/vector if interflow-qpos	x			ACD; Look-Ahead Interflow[1]
goto step/vector if queue fail			x	
goto step/vector if holiday in/not-in table	x		x	Holiday Vectoring
messaging split/skill	x	x		
messaging split/skill active/latest[2]	x	x		
queue-to best	x			ACD; G3V4 Advanced Routing; Best Service Routing
queue-to split/skill	x			ACD
queue-to attd-group				Attendant Vectoring
queue-to attendant				Attendant Vectoring
queue-to hunt group				Attendant Vectoring
reply-best	x			ACD; G3V4 Advanced Routing; Best Service Routing; Look-Ahead Interflow[1]
route-to number	x			
route-to digits with cov y (n)		x		
route-to number if digit		x		
route-to number if unconditionally with cov y (n)[2]	x	x		

Vector commands / required options summary (continued)

Command	Basic	Prompting	Attendant	Other Options Required
route-to number if digit with cov y (n)[2]		x		
route-to number if unconditionally	x	x		
route-to number if interflow-qpos	x			ACD, Look-Ahead Interflow[1]
stop	x	x		
wait-time <time>	x	x	x	
wait-time <time> hearing <treatment>	x	x	x	
wait-time <time> hearing <extn> then <treatment2>	x	x	x	

1. Provided with Virtual Routing RTU (right to use).

2. If G3V4 software has not been purchased, these commands require the G3V4 maintenance load.

Command description/reference

The following table provides a brief description of the function of each of the Call Vectoring command. See the listed page number for a complete description of the command.

Command description/reference table

Command	Function
Adjunct routing command on page 403	To request adjunct to route call.
Announcement command on page 410	To connect caller to delay recording.
Busy command on page 412	To connect caller to busy tone.
Check command on page 414	To connect/queue call on a conditional basis.
Collect Digits command on page 418	To prompt caller for digits.
Consider command on page 423	To obtain BSR status data from a local split/skill or a remote location
Converse-on command on page 428	To deliver a call to a converse split/skill and to activate a Voice Response Unit (VRU).
Disconnect command on page 437	To force disconnect of call with optional announcement.
Goto step and goto vector commands on page 439	To cause unconditional/conditional branch to another step in the vector.
Messaging command on page 446	To allow caller to leave message for callback.
Queue-to command on page 449	To connect/queue call to the primary split/skill or to the best resource (or, with Attendant Vectoring, attendant, attendant group, or hunt-group) found by a consider series.
Reply-best on page 453	To send BSR status data to the primary vector in a multi-site application
Route-to command on page 455	To connect call to destination entered via **`collect digits`** command, or to connect call to internal/external destination.
Stop command on page 463	To stop further vector processing.
Wait-time command on page 464	To initiate feedback to caller (if needed) and delay processing of the next step.

Command job aid

The vector command job aid shown below lists the Call Vectoring commands, together with the various conditions, and parameter options and values that are available for use with each command.

Obtaining switch capacity information

Most vector commands require one or more input values for the command, as well as for various parameters, such as an announcement extension number, a time interval, a maximum queue size, and so forth. When the minimum and maximum ranges for command parameter values are identical for all Avaya switch platforms, the limiting ranges are specified in the job aid. Alternately, when the minimum and maximum ranges for a parameter value are not the same among the among Avaya switch platforms, the upper limit of a value range is indicated by the term “switch max.”

To determine the maximum values you can use in Call Vectoring commands, see *System Capacities Table for Avaya MultiVantage on Definity Servers*, 555-233-605. To access the document online, go to:

http://avayadocs.com

Vector command job aid

Command	Parameters / Conditions		
adjunct routing link ___ CTI-link ID 1 to 16			
announcement ________ extension no.			
busy			
check			
best	**if ______** ________ expected wait < 1 to 9999 seconds unconditionally wait improved > 0 to 9999 seconds		
skill _____ 1 to switch max or 1st, 2nd, 3rd VDN skill (**EAS-only**)	**pri ___** **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	**if ________** available-agents **>** 0 to 1499[1] calls-queued **<** 1 to 999[1] expected-wait **<** 1 to 9999 seconds oldest-call-wait **< >**1 to 999 seconds rolling-asa **<** 1 to 999 seconds staffed-agents **>** 0 to 1499[1] wait-improved **>** 0 to 9999 seconds unconditionally	
split _____ 1 to switch max	**pri ___** **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	**if ________** ___ ___________ available-agents **>** 0 to 1499[1] calls-queued **<** 1 to 999[1] expected-wait **<** 1 to 9999 seconds oldest-call-wait**<=>** 1 to 999 seconds rolling-asa **<** 1 to 999 seconds staffed-agents **>** 0 to 1499[1] wait-improved **>** 0 to 9999 seconds unconditionally	
collect ____ digits			
ced or cdpd			
1 to 16 digits	**after announcement _____** extension no. or “none”		

Vector command job aid (continued)

Command	Parameters / Conditions	
consider ____		
location ____ 1 to 255 (**multi-site BSR only**)[2]	**adjust by** ____ 0-100 percent	
skill _____ 1 to switch max or 1st, 2nd, 3rd VDN skill (**EAS only**)	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	**adjust by** ____ 0- 100 percent
split _____ 1 to switch max	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	**adjust by** ____ 0- 100 percent
converse-on		
skill _____ 1 to switch max 1st, 2nd, 3rd VDN skill (**EAS only**)	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	**passing** ____ **and** ____ 6-digit string, "*" , "#" , "none"[3], "ani", "vdn", "digits", "qpos", "wait"
split _____ 1 to switch max	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	**passing** ____ **and** ____ 6-digit string, "*" , "#" , "none"[3], "ani", "vdn", "digits", "qpos", "wait"
disconnect	**after announcement** _____ extension no. or "none"	

Vector command job aid (continued)

Command		Parameters / Conditions		
goto				
step ____ 1-32 or **vector** ____ 1 to switch max				
if				
ani ___ <,>,<=,>=,<>,=		____ 1 to 16 characters including digits (0-9), "?", "+" "none"[4]		
in, not-in		**table** ____ 1 to 100[1]		
available-agents				
	in skill ____ 1 to switch max, or 1st, 2nd, or 3rd for VDN Skill		____ <,<= >,>=,<>,=	____ 1 to 1500[1] 0 to 1499[1]
	in split ____ 1 to switch max		____ <,<= >,>=,<>,=	____ 1 to 1500[1] 0 to 1499[1]
calls-queued				
	in skill ____ 1 to switch max, or 1st, 2nd, or 3rd for VDN Skill (**EAS-only**)	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	____ <,<= >,>=,<>,=	____ 1 to 999[1] 0 to 998[1]
	in split ____ 1 to switch max	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	____ <,<= >,>=,<>,=	____ 1 to 999[1] 0 to 998[1]
counted-calls		**to vdn** ____ vdn extension, "latest" or "active"[5]	____ <,<= >,>=,<>,=	____ 1 to 999[1] 0 to 998[1]
digits ___		____		
	<,>,<=,>=,<>,=	1-16 digits, including 0-9, "#", "?", "+", or "none"[4]		
	=	meet-me-access[6]		
	in, not-in	**table** ____ 1 to 100[1]		

Vector command job aid (continued)

Command	Parameters / Conditions			
goto step (or vector) (*continued*)				
if (*continued*)				
expected-wait				
for ___ best, call	____ <,>,<=,>=,<>,=	____ 0-9999 seconds		
for ___ split	____ 1 to switch max	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	____ <,>,<=,>= <>,=	____ 0 to 9999 sec.
for ___ skill (**EAS-only**)	____ 1 to switch max, 1st, 2nd, 3rd skill for VDN	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	____ <,>,<=,>= <>,=	____ 0 to 9999 sec.
holiday ____ in, not-in	**table** ____ 1-10			
ii-digits	____ <,>,<=,>= <>, =	____ 2-digit string, “+”, “?”, or “none”[4]		
	in, not-in	**table** ____ 1 to 100[1]		
interflow-qpos	____ <,>,<=,>= <>, =	____ 1-9		
meet-me-full[6] **(go-to step, only)**				
meet-me-idle[6] **(go-to step, only)**				
no match[7]				
oldest-call-wait				
	in skill _____ 1 to switch max, or 1st, 2nd, 3rd, skill for VDN (**EAS-only**)	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	____ <,>,<=,>= <>, =	____ 0 to 999 seconds
	in split ____ 1 to switch max	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	____ <,>,<=,>=, <>,=	____ 0 to 999 seconds

Vector command job aid (continued)

Command		Parameters / Conditions			
goto step (or vector) (*continued*)					
if (*continued*)					
queue-fail[8]					
rolling-asa	**for** ____	skill (**EAS-only**)	______ 1 to switch max, or 1st, 2nd, 3rd skill for vdn	____ <,>,<=,>= <>, =	____ 0-999 seconds
		split	______ 1 to switch max	____ <,>,<=,>= <>, =	____ 0-999 seconds
		vdn	______ vdn extension, “latest” or “active”[5]	____ <,>,<=,>= <>, =	____ 0-999 seconds
staffed-agents		**in skill** ______ 1 to switch max, or 1st, 2nd, 3rd skill for vdn		____ <,>,<=,>=, <>,=	____ 1 to 1500[1]
		in split ______ 1 to switch max		____ <,>,<=,>=, <>,=	____ 1 to 1500[1]
time-of-day	**is**	____ ____ ____ mon-sun, “all” 00-23 hour 00-59 minute	**to**	____ ____ ____ mon-sun, “all” 00-23 hour 00-59 minute	
wait-improved for		______ best	________ <,>,<=,>= <>, =	______ 0 to 9999 seconds	
		skill ____ 1 to switch max, or 1st, 2nd, 3rd skill for vdn (**EAS-only**)	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	_____ <,>,<=,>=, <>,=	____ 0-9999 seconds
		split ____ 1 to switch max	**pri** ____ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	____ <,>,<=,>=, <>,=	____ 0-9999 seconds
unconditionally					

Vector command job aid (continued)

Command	Parameters / Conditions	
messaging		
	skill ____ 1 to switch max, or 1st, 2nd, 3rd skill for vdn (**EAS-only**)	**for extension** ____ extension no., “latest” or “active”[5]
	split ____ 1-switch max	**for extension** ____ extension no., “latest” or “active”[5]
queue-to		
	attd-group[8]	
	attendant[8] _________ extension no.	
	best	
	hunt-group[8] _____ 1 to switch max	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority
	skill ____ 1 to switch max, or 1st, 2ns, 3rd skill for vdn (**EAS-only**)	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority
	split ____ 1 to switch max	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority
reply-best (**multi-site BSR only**)[2]		

Vector command job aid (continued)

Command	Parameters / Conditions			
route-to				
	digits	**with coverage** _____ **y**es or **n**o		
	meetme[6]			
	number ____ 0-9, * , # , ~p, ~m, ~s, ~w, ~W, or a leading ~r[9]	**with cov** ___ **y**es or **n**o	**if** ____ digit	
			interflow-qpos	___ ____ <, =, <= 1-9
			unconditionally	
	name1 [7]	**with coverage** _____ **y**es or **n**o		
	name2 [7]	**with coverage** _____ **y**es or **n**o		
	name3 [7]	**with coverage** _____ **y**es or **n**o		
stop				
wait-time				
____	____	**hearing**	_____________	
0-999 0-480[10] 0-8[10]	secs mins hrs		music, ringback, silence, i-silent	
			or audio source ext.	**then** __________ music, ringback, silence, continue[11]

1. Maximum limit less on some platforms. Use the help key for your switch administration software to determine the applicable limit for your system.
2. This item available with the Virtual Routing feature, only.
3. If the first "passing" value is entered as "**none**" the second value must also be entered as "**none**."
4. The question mark (**?**) is a wild card that matches any digit (0-9) at the specified position. The plus sign (**+**) matches any or no characters at the specified position.
5. "**Active**" refers to the VDN specified by VDN Override settings. "**Latest**" refers to the VDN specified for the current vector.
6. This item available with **meet-me conference** vectors, only.
7. This item available with Dial by Name feature, only.
8. This item available with Attendant Vectoring feature, only.
9. When the specified number is preceded by **~r**, Network Call Redirection is attempted.
10. This option is not available for vector administration done through Avaya CMS or Visual Vectors.
11. **Continue** is only a valid treatment with Multiple Audio/Music Sources. It indicates that the caller will continue to hear the alternate audio/music source (using an announcement) until another vector command takes effect.

Command directory

The manual page directory in this section lists and discusses all of the Call Vectoring commands:

Adjunct routing command

Purpose

The **adjunct routing** command causes a message to be sent to an adjunct requesting routing instructions.

Syntax and valid entries

Command	Parameters / Options
adjunct routing	**link __** CTI-link ID (1-16)

Requirements

ASAI software must be installed.

An ISDN-BRI or MAPD port is required, and the port must be connected to an ASAI host.

Operation

The **adjunct routing** command provides a means for an adjunct ASAI processor to specify the destination of a call. The switch provides information in an ASAI route request message that the ASAI adjunct can use to first access a data base and then determine a route for the call. In a typical application, the ASAI adjunct might use the dialed number, the calling party number (CPN/BN), or the digits collected via Call Prompting or Caller Information Forwarding (CINFO) to access customer information and thereby determine the call route. A maximum of 16 digits collected from the last **collect digits** command can be passed.

An adjunct specified in an **adjunct routing** command can route a call to an internal number, an external number, a split, a VDN, an announcement extension, or a particular agent. An adjunct can also provide priority ringing, priority queuing, and specify that a route to an agent be done as a direct agent call.

When a call encounters an **adjunct routing** command, the switch sends to the specified adjunct an ASAI message requesting a call route. The following list identifies the contents of the message, along with a comment or a brief explanation for each item:

- **Calling number information**. Calling party number or billing number (CPN/BN) provided by ISDN-PRI or R2MFC signaling facilities. If the call originates from a local switch extension, this extension is the calling number.

- **Originating line information (II-digits)**. Two-digit code provided by ISDN-PRI facilities indicating the type of originating line being used.
- **Called number**. Originally called extension (if a call is forwarded to a VDN), or the first VDN through which the call was routed (if the call was not forwarded to the VDN).
- **Routing VDN**. Last VDN that routed the call to the vector that contains the **`adjunct routing`** command.
- **Call identifier**. ASAI identifier that permits the ASAI adjunct to track multiple calls via either Event Notification or Third Party Call Control. See *Avaya MultiVantage CallVisor ASAI Technical Reference*, 555-230-220 for more information on ASAI.
- **Look-Ahead Interflow (LAI) information** (if any). Includes the original VDN display information, the priority level of the call at the originating switch, and the time that the call entered vector processing.
- **Digits collected via Call Prompting** (if any). Digits are collected by the most recent **`collect digits`** command. These could be CINFO digits, but if so it will not be indicated by ASAI. See Call Prompting on page 181 for more information.
- **User-to-User Information** (if any). ASAI user-provided data associated with the call. If provided by ASAI, this data was provided in a 3rd-Party-Make-Call, Auto-Dial, or Route-Select message. If provided over ISDN, the data was in the SETUP message that delivered the call to this switch.

The **`wait-time hearing i-silent`** command is used in cases where it is important to allow the adjunct to decide whether to accept an incoming ISDN-PRI call. When this step is encountered after an **`adjunct routing`** step, the switch does not return an ISDN PROGress message to the originating switch. This is particularly important for Network ISDN features and for the LAI feature.

If the call is queued, the **`adjunct routing`** step is ignored, and vector processing continues at the next vector step.

If the ASAI link specified in the **`adjunct routing`** step is down, the step is skipped.

An ASAI link failure can change the manner in which subsequent treatment (that is, **`announcement`** and/or **`wait-time`**) steps (if any) in the vector are usually processed. In some cases, such processing is influenced by the position that the treatment steps occupy in the vector. In other cases, the positioning of these commands along with their relationship to specific **`goto`** commands come into play. For example, any **`announcement`** or **`wait-time`** step that immediately follows an **`adjunct routing`** step whose ASAI link is down is skipped.

The second step after the **`adjunct routing`** step is often implemented as a default treatment (for example, a route-to an attendant). If the ASAI link is down, the default step executes immediately. Otherwise, the step executes only if the application does not respond with a route within the time period specified by the **`wait-time`** step.

On the other hand, if a **goto** step follows an **adjunct routing** step, the switch executes the **goto** step and then skips various treatment steps according to their position in the vector, and the conditional determination of the **goto** step. Specifically, if the **goto** step succeeds and the branch is taken, the switch skips any **announcement** or **wait-time** step that is the first non-**goto** step branched to by the **goto** step.

Note:

The first step to which a **goto** step is usually designed to branch (other than another **goto** step) is a nontreatment step. That is, a step containing a command other than a **wait-time** or an **announcement** command).

Alternately, if the **goto** step fails and the branch is not taken, the switch skips any **announcement** or **wait-time** step that immediately follows the **goto** step if the application is down.

Note:

The **goto** step that fails can be at the end of a sequence of **goto** steps that branch to each other.

After the switch sends a route request to the ASAI adjunct, vector processing continues with the vector steps that follow.

The step that follows the **adjunct routing** step, in effect, determines the maximum length of time the switch will wait for the ASAI adjunct to reply with a call route. Accordingly, you should always include either a **wait-time** step or an **announcement** step immediately after an **adjunct routing** step. Moreover, the switch cancels the route request if vector processing encounters a step containing any of the following commands:

- **busy**
- **check split**
- **collect digits**
- **converse-on split**
- **disconnect**
- **messaging split**
- **queue-to split**
- **route-to**

Note:

Multiple adjunct routing steps can follow each other in sequence. Each step activates a separate adjunct route request. Any intervening vector commands (or blank steps) between two adjunct routing commands cancels any previous route-to requests.

If a valid call route is received by the switch via a route-select message before one of the vector commands in the previous list is executed, the switch routes the call to the destination specified by the adjunct route. Otherwise, the route request is terminated without affecting vector processing.

The adjunct can also decide to not route a call by rejecting (negatively acknowledging) the route request sent by the switch, or the link/application can go down. Upon receiving a route request rejection, or detection of a link/application failure, the switch terminates the **`announcement`** or **`wait-time`** step that is being executed for the call and then continues with the next vector step.

When the switch receives a call route (route-select to a destination) from the ASAI adjunct, the switch first validates the route as follows:

1. The switch verifies that the VDN's COR permits the call to be terminated at the adjunct-supplied destination.
2. The switch verifies that the adjunct-supplied information (destination number, ACD split, TAC/AAR/ARS access code, etc.) for the route is valid. This includes checking that the destination is compatible with the dial plan, and that the options specified by the adjunct are correct.
3. If the ASAI adjunct specifies the Direct Agent Call (DAC) option, the destination number (agent) must be logged into the adjunct-specified ACD split.
4. If the destination for the call is external, the switch verifies the trunk is available for the call.

If any of these conditions are not met, the route validation fails, and the switch does the following:

1. Discards the route.
2. Notifies the ASAI adjunct that the route is invalid.
3. Continues with vector processing.

If the route is valid, the switch does the following:

1. Terminates vector processing immediately.
2. Notifies the ASAI adjunct that the route is accepted.
3. Routes the call to the destination specified by the ASAI adjunct.

When the call is routed, the caller hears normal call progress tones and feedback. However, if the call is routed to an extension with no available call appearances and no coverage path, the caller hears the busy tone. Any other features that may be in effect at the adjunct-supplied destination (such as Send-All-Calls or Call Forwarding) interact with the routed call.

Note:

The operation described above is similar to that for the **`route-to with coverage set to yes`** commands.

Answer supervision considerations

The command has no interaction with answer supervision.

If adjunct routing is used with ISDN-PRI, then an **adjunct routing** command followed by a **wait-time hearing silence** signals the originating switch that the receiving switch has accepted the call (for Lookahead Interflow), even though answer supervision has not been provided. To prevent this from occurring, use the **wait-time hearing i-silent** option after the **adjunct routing** step.

Feature interactions

For a call coming in directly to a VDN, the command is treated like a **route-to** command that has the **with cov** or **with coverage** parameter set to **y**.

Note:

If the Display VDN for Route-to DAC option is enabled for the VDN, the name of the VDN is displayed at the agent station for a call that is routed through an adjunct. For more information, see Displaying VDN names for vector-initiated Direct Agent calls on page 480.

For a call that is covered to a VDN, the command is treated like a **route-to with coverage=n** command. A covered call that is routed by an **adjunct** routing command to a destination that has Call Forwarding activated is not further redirected (since the call has already been redirected by coverage).

For LAI or Network ISDN features, the adjunct routing command is considered a neutral vector command in all cases. However, the command is usually followed by an **announcement** or **wait-time** command, each of which is a call acceptance command. The G3V4 **wait-time hearing i-silent** command can be used when a neutral **wait-time** command is required to allow the adjunct to accept or reject the call.

If an **announcement** command follows a failed **adjunct routing** command, the announcement is interrupted. If the **adjunct routing** command succeeds (that is, the switch receives a destination from the ASAI adjunct), the announcement terminates immediately.

If an ASAI adjunct has supplied dial-ahead digits for a **collect digits** step, and the vector processes a **collect ced digits** or **collect cdpd digits** step, the ASAI supplied dial-ahead digits are discarded without notification to the adjunct.

If a TTR is connected to a call because an ASAI adjunct has requested digit collection, and the vector processes a **collect ced digits** or **collect cdpd digits** step, the TTR is disconnected from the call.

CMS interactions

Adjunct routing attempts are stored in the ADJATTEMPTS database item and reported as Adjunct Routing Attempts in standard reports. If the call is queued to a split/skill when the **`adjunct routing`** command is encountered, the step is skipped, and no messages are sent to CMS. Accordingly, Adjunct Routing Attempts is not reported for this call.

When a routing response from the adjunct is successfully executed by the switch, this action is tracked in the ADJROUTED and ADJROUTTIME database items and shown as Adjunct Routing Completions in standard reports.

Additional tracking of the **`adjunct routing`** command varies based on the destination successfully routed to as follows:

Routed to station or to attendant		
Database item	Report Heading	Notes
OUTFLOWCALLS/ OUTFLOWTIME	Vector Flow Out	
INTIME	Avg Time In Vector	
CONNECTCALLS/ CONNECTTIME	Other Calls Connect	answered calls on R5

Routed to trunk		
Database item	**Report heading**	**Notes**
OUTFLOWCALLS/ OUTFLOWTIME	Vector Flow Out VDN Flow Out	
INTERFLOWCALLS/ INTERFLOWTIME	VDN Flow-Interflow	
INTIME	Avg Time In Vector	

Routed to VDN		
Database item	**Report heading**	**Notes**
OUTFLOWCALLS/ OUTFLOWTIME	Vector Flow Out VDN Flow Out	
INTIME	Avg Time In Vector	
INFLOWCALLS	Vector Flow In VDN Flow In	new vector new VDN

Routed to split or to hunt group		
Database item	**Report heading**	**Notes**
CALLSOFFERRED		new split
LOWCALLS/MEDCALLS		no priority/priority

Split/skill calls are also shown in the standard reports based on the final disposition of the call.

The presence of the command in a vector enables the calls serviced by the vector to be vector-directed. When such a call is answered by an agent, the call is tracked as ACDCALLS/ANSTIME, and it is reported as ACD Calls, Split/skill ACD Calls, and Avg Speed Ans.

A call abandoned after the command routes the call to a station or an attendant is tracked in the VDN tables as ABNCALLS/ABNTIME.

BCMS interactions

If the command advances a call to another position (that is, ASAI routing is successful), the call is tracked as outflow in the VDN Report.

Announcement command

Purpose

Provides the caller with a recorded announcement.

Syntax and valid entries

```
announcement________
       valid extension no.
```

Requirements

Integrated board, aux trunk or analog (T&R or Lineside DS1) announcement equipment must be installed.

Appropriate announcements need to be administered and recorded. For more information, see “Managing Announcements” in *Administrator Guide for Avaya MultiVantage Software*, 555-233-506.

Operation

The announcement is played from beginning to end unless an agent becomes available. In such a case, the announcement is interrupted and (if manual answering operation is assigned to the agent, or if calls are delivered to the agent on a manual answering basis) ringback is provided. If the call is queued, the call remains as such while the announcement is played. Any feedback that is provided before an announcement (for example, a wait with music or ringback) continues until the announcement is played.

If the announcement’s queue is full, the call retries the announcement step for an indefinite period of time before any new vector steps are processed.

If an **`announcement`** command follows a failed **`adjunct routing`** command, the announcement is interrupted. If the **`adjunct routing`** command succeeds (that is, the switch receives a destination from the ASAI adjunct), the announcement terminates immediately.

The **announcement** command step is skipped, and vector processing continues at the next vector step, whenever any of the following conditions exist:

- Requested announcement is busied out, not available, or not administered.
- Integrated board is not installed.
- External aux trunk or analog equipment is not attached.

For a complete description of the types and operation of announcements see “Managing Announcements” in *Administrator Guide for Avaya MultiVantage Software*, 555-233-506.

Answer supervision considerations

Unless answer supervision has already been sent, it is sent as soon as the command starts to process the call (even before the announcement starts).

Feature interactions

For LAI, the command may be considered a call acceptance vector command or a neutral vector command.

The command is considered a call acceptance vector command whenever one of the following is true:

- Announcement is available.
- Call is queued for an announcement.
- Announcement is retried.

The command is considered a neutral vector command whenever the announcement is unavailable.

CMS/BCMS interactions

The command is not tracked by CMS or BCMS.

Busy command

Purpose

The **busy** command gives the caller a busy signal and causes termination of vector processing.

Syntax

```
busy
```

Requirements

Operation

The command takes effect on non-CO trunk calls whether or not answer supervision has been sent. However, if the call is on a CO trunk and answer supervision has not been sent, the busy is not passed back by the CO, and the caller continues to hear ringback from the CO. Calls are dropped approximately 45 seconds after the busy tone is applied.

If ISDN-PRI is involved, the application of the busy tone is enabled via D-channel messaging. The network switching office returns the busy tone to the caller. The facility to the switch is dropped, thus making it immediately available for another call.

Answer Supervision Considerations

After the 45 second timeout, an unanswered CO trunk call is answered and then dropped. All other unanswered calls after this timeout are dropped without being answered. For an ISDN call that has not yet queued or been answered, no timeout occurs, and answer supervision is not sent. Instead, a message requesting a busy tone is sent to the network and, subsequently, the trunk is released.

Feature interactions

For LAI or BSR, the command is considered a call denial vector command in all cases.

CMS interactions

Busy command	
Database Item	**Report Heading**
BUSYCALLS/BUSYTIME	Calls Forced Busy Calls Busy/Disc
OTHERCALLS/OTHERTIME	Inbound Other Calls
INTIME	Avg Time In Vector

BUSYTIME, OTHERTIME, and INTIME for splits and vectors are tracked according to when the busy tone starts. BUSYTIME, OTHERTIME and INTIME for VDNs are tracked according to when the trunk idles.

BCMS interactions

A call that is forced busy due to the command is tracked as OTHER in the VDN Report.

Check command

Purpose

Checks the status of a split/skill for possible termination of the call to that split/skill.

Syntax and valid entries

Command	Parameters / Conditions		
check			
best	**if** ______ expected wait unconditionally wait improved	_________ < 1 to 9999 seconds > 0 to 9999 seconds	
skill _____ 1 to switch max or 1st, 2nd, 3rd VDN skill (**EAS-only**)	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	**if** ________ available-agents calls-queued expected-wait oldest-call-wait rolling-asa staffed-agents unconditionally wait-improved	 **>** 0 to 1499[1] **<** 1 to 999[1] **<** 1 to 9999 seconds **<** **>** 1 to 999 seconds **<** 1 to 999 seconds **>** 0 to 1499[1] **>** 0 to 9999 seconds
split _____ 1 to switch max	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	**if** ________ available-agents calls-queued expected-wait oldest-call-wait rolling-asa staffed-agents unconditionally wait-improved	___ ___________ **>** 0 to 1499[1] **<** 1 to 999[1] **<** 1 to 9999 seconds **<=>** 1 to 999 seconds **<** 1 to 999 seconds **>** 0 to 1499[1] **>** 0 to 9999 seconds

1. Maximum limit less on some platforms. Use the help key for your switch administration software to determine the applicable limit for your system.

Requirements

Operation

The **`check`** command checks the status of a split/skill against conditions specified in the command. If the conditions specified in the command are met, the call is terminated to the split/skill. If the conditions are met but no agents are available, the call is queued to the split/skill and waits for an agent to become available.

Each **`check`** command may be used with the keyword one of the following three keywords: **`split`**, **`skill`**, or **`best`**. The **`check split`** or **`check skill`** command requires you to specify the split/skill to be checked. The **`check best`** command checks the status of the best split/skill identified by the immediately preceding series of **`consider`** steps, then either terminates or queues the call to that split/skill. You don't have to specify the split/skill in **`check best`** commands since the switch compares two or more skills and identifies the "best" in the preceding series of **`consider`** steps.

The command is customized to check for and/or respond to specific conditions. For example, the command can queue/terminate unconditionally. The command can also queue/terminate if any of the following is true:

- Number of available agents is greater than the threshold value.
- Number of staffed agents is greater than the threshold value.
- Number of calls queued for a specified priority level or higher is less than the threshold value.
- Oldest call waiting in queue at the specified priority level or higher has been waiting less than the threshold value, which is expressed in seconds.
- Rolling average speed of answer is less than the threshold value, which is expressed in seconds.
- Expected wait time is less than the threshold value, which is expressed in seconds.
- Expected wait time will be improved by more than the threshold value, which is expressed in seconds, by queuing the call to the split/skill specified. EWT in the specified split/skill is compared to the call's current EWT. (A call's EWT will be infinite if the call is not in a queue.)

A call may be queued to up to three splits/skills simultaneously. A call remains queued either until vector processing terminates (via a successful **`disconnect`**, **`busy`**, or **`route-to`** command, or via an abandoned call), the call is routed to another VDN (by a **`route-to number`** or **`route-to digits`** command), or the call reaches an agent. When an agent becomes available in any split/skill to which the call is queued, the following actions take place:

- Call begins ringing the agent.
- Call is removed from any other queues.
- Vector processing terminates.

If the desired backup split/skill is one of the splits/skills to which the call is already queued, the call is requeued at the new priority level, provided that the command conditions are met. The step is skipped, and vector processing continues at the next step if any of the following conditions are true:

- Command conditions are not met.
- Desired split's (skill's) queue is full.
- Desired split/skill has no queue and also no available agents.
- Desired split/skill is not vector-controlled.
- Call is already queued to this split/skill at the specified priority level.
- Call has been previously queued to three different splits/skills.

Note:

A **`route-to`** to another VDN can be used to remove the call from the splits it is queued to if necessary. The steps in the routed-to vector then can be used to queue to other splits.

Answer Supervision Considerations

No answer supervision is returned.

Feature interactions

The **`check`** command can access an AUDIX/Message Center/Server split/skill in cases where a VDN is assigned as a coverage point. To enable this function, the split/skill must be assigned as a vector-controlled hunt group.

For BSR and LAI, the command can be considered either a call acceptance vector command or a neutral vector command. For more on BSR interactions, see Best Service Routing (BSR) on page 229.

The command is considered a call acceptance vector command whenever one of the following is true:

- Call terminates to an agent.
- Call queues to a split/skill.
- BSR interflowed call is accepted at remote interflow vector.

The command is considered a neutral vector command when the call neither terminates nor queues.

No COR checking is carried out when a **`check`** step places a call to a split/skill.

The **`oldest-call-waiting`** condition can check only priority level l (low).

CMS interactions

Calls answered via the check command are indicated as answered by backup in CMS.

Calls queued via a **`check split/skill`** command are tracked as CALLSOFFERRED and LOWCALLS/MEDCALLS/HIGHCALLS/TOPCALLS.

The presence of the command in a vector enables the calls serviced by the vector to be vector-directed. When such a call is answered by an agent, the call is tracked as ACDCALLS/ANSTIME, and it is reported as ACD Calls, Split/Skill ACD Calls, and Avg Speed Ans. If the call is also queued to other splits/skills, OUTFLOWCALLS/OUTFLOWTIME is tracked in the first split/skill to which the call queues, and Flow Out is reported (unless the split/skill turns out to be the answering split/skill). DEQUECALLS/DEQUETIME is tracked in the second and third splits/skills if these splits/skills are not the answering split/skill, and the call is reported as Dequeued Calls and Dequeued Avg Queue Time. However, if the second or third split/skill is the answering split/skill, INFLOWCALLS is tracked in the split/skill, and the call is reported as Flow In.

Whenever the call is answered in a split/skill accessed by the **`check split/skill`** command, the BACKUPCALLS data base item is incremented, and the call is reported as Calls Ans in Backup and Calls Handled/Backup. The Calls Ans in Main report item is calculated by using the algorithm ACDCALLS - BACKUPCALLS.

If the call abandons after the command queues the call to a split/skill, ABNCALLS/ABNTIME is tracked for the vector, the VDN, and the first split/skill to which the call is queued. The call is reported as Aban Call and Avg Aban Time. If the call is also queued to other splits/skills, DEQUECALLS/DEQUETIME is tracked in these splits/skills, and the call is reported as Dequeued Calls and Dequeued Avg Queue Time.

BSR status poll calls are not counted as interflows. BSR interflows are now tracked as network interflowed calls (NETCALLS) by the CMS at the receiving switch. The CMS tracks a call’s accumulated time-in-VDN as NETINTIME (that is, the NET_TIME value on the CMS at switch C combines the time a call has spent in VDNs at any previous locations, as communicated by ISDN information forwarding. The NETINTIME can be added to the time spent in the local switch to provide reports that include the total time the call has spent in the call center network (e.g., total ASA).

For more information on CMS database items and reports, see *Avaya CMS Database Items and Calculations*, 585-780-702, and *Avaya Call Management System Supervisor Version 11 Reports*, 585-210-708.

BCMS interactions

The total number of calls to the VDN that are queued via the command and then answered by an agent within a specified time period is tracked as ACD Calls in the VDN Report. The average time that calls spend in a vector before being connected via the command as an ACD call to an agent is tracked as AVG SPEED ANS in the same report.

There is no added tracking for calls interflowed by BSR. BCMS tracks these calls as outflow in the VDN Report.

Collect Digits command

Purpose

The **collect digits** command allows the user to enter up to 16 digits from a touch-tone phone or an internal rotary phone, or allows the vector to retrieve Caller Information Forwarding (CINFO) digits from the network.

Syntax and valid entries

Command	Parameters / Conditions
collect ____ digits	
1-16 digits	**after announcement _____** extension no. or “none”
ced or cdpd	

Requirements

The Avaya Call Center Deluxe package or Avaya Call Center Elite package must be installed. This command is also available with the Automated Attendant RTU.

At least one TN744 Call Classifier circuit pack or TN2182 Tone Clock circuit pack must be in the system unless the command is used only to collect digits returned by a VRU or sent by the network and never to collect digits from a caller.

The Vectoring (CINFO) feature used to collect ced or cdpd digits from the network ISDN and the AT&T Network Intelligent Call Processing (ICP) service or equivalent.

Operation

The collect command has twomodes of operation:

- Collecting digits on the switch
- Collecting CINFO digits

Collecting Digits on the switch – The **`collect digits`** command allows a caller to enter digits from a touch-tone or an internal rotary phone. An optional announcement may be used to request the caller to enter these digits. The announcement can instruct the user to enter an asterisk (*) if incorrect data is entered. When the caller enters an asterisk, the digits collected for the current **`collect digits`** command are deleted, digit collection is restarted, and the announcement is not replayed.

Note:

You can set the `Reverse Star/Pound Digit For Collect Step?` field on the ISDN Parameters page of the Feature-Related System Parameters form to **y** in order to reverse the normal handling of the asterisk (*) and pound (#) digits by the `collect` vector command. With the Reverse Star/Pound Digit for Collect Step set to **y**, the asterisk (*) digit is interpreted as a caller end-of-dialing indicator and the pound (#) digit is interpreted to clear all digits that were previously entered for the current collect vector step.

In using this command, the maximum number of digits requested of the caller must be specified in the administration of the command. If the caller can enter fewer digits than the maximum specified, the announcement should instruct the caller to terminate the entry with a pound sign (#) digit as an end-of-dialing indicator. If all the digits strings for all the variations of a specific **`collect digits`** command are terminated with **#**, the **#** must be counted as one of the digits. Therefore, the number of digits collected should include any **#** that needs to be collected. Otherwise, the terminating **#** is kept as a dial-ahead digit and is processed by a subsequent **`collect digits`** command. If fewer digits than the maximum specified are entered, and if the caller does not complete the entry with a pound sign, an interdigit timeout occurs. The timeout terminates the command, and any digits collected prior to the timeout are available for subsequent vector processing.

Generally, processing of the command requires that a TTR be connected. (If the call originates from an internal rotary phone, no TTR is needed.) TTRs accept the touch-tone digits that are entered by Call Prompting users. TTRs are automatically connected as needed by the system.

The connection of the announcement prompt is skipped and the digit collection phase begins whenever one of the following conditions is true:

- Dial-ahead digits exist.
- No announcement is administered for the **`collect digits`** step.
- Announcement administered for the **`collect digits`** step does not exist.

Otherwise, an attempt is made to connect the administered announcement. If the announcement to be connected is busy, and if the queue for the announcement is full, or if there is no queue, the calling party continues to hear the current feedback. The system waits five seconds and then tries again to connect the call to the announcement. This process continues until the call is successfully queued or connected to the announcement, or until the calling party disconnects from the call. If the queue for the announcement is not full, the call is queued for the announcement.

If the announcement to be connected is available (either initially or after queuing, or after system retry), any previous feedback is disconnected, and the calling party is connected to the announcement.

While the announcement is playing, or while the call is being queued for an announcement, the caller may enter digits at any time. This causes the announcement to be disconnected or removed from the queue, as appropriate, and the digit collection phase to begin. If the caller does not enter any digits during the announcement phases, the digit collection phase begins when the announcement completes.

As soon as the digit collection phase begins, interdigit timing is started, unless the TTR is already in timing mode (that is, the dial-ahead capability is active and the TTR is not disconnected).

Digits are collected either as digits dialed during the **`collect digits`** command or as dial-ahead digits dialed since a previous **`collect digits`** command but prior to the current appearance of the command. Digit collection continues for the current command until one of the following conditions exists:

- Number of digits specified is collected.
- Pound sign (#) digit is collected (signifying end of dialing).
- Inter-digit timer expires.

If, during the digit collection phase, a **`*`** is encountered within a stream of dialed or dial-ahead digits, all digits that are collected for the current **`collect digits`** step are discarded. If additional dial-ahead digits occur after the asterisk, these digits continue to be processed. If there are no such digits, and if no TTR is connected, vectoring continues at the next vector step. If a TTR is connected, the caller can start entering digits again. In such a case, the announcement is not replayed, and the interdigit timer is restarted.

Note:

> If an asterisk is entered after the requested number of digits are entered, the asterisk has no effect on the previously entered digits. However, in such a case, the asterisk is treated as a dial-ahead digit for the next **`collect digits`** command.

When digit collection is completed, and if a TTR is connected (for a touch-tone phone), the interdigit timer is restarted to detect a timeout for releasing the TTR. Vector processing then continues at the next vector step. However, the switch continues to collect any subsequent dialed digits (including the pound sign (#) and asterisk (*) digits) to allow for the dial-ahead capability. These additional "dialed ahead" digits are saved for use by subsequent **`collect digits`** commands, and they provide the caller with a means to bypass subsequent unwanted announcement prompts. A single # digit can be collected and tested by subsequent **`route-to...if digits`** or **`goto...if digits`** commands. Alternately, any collected digits (whether collected from callers or CINFO) can be passed to a host via ASAI or forwarded to another site via Information Forwarding. Collection of dial-ahead digits continues until one of the following occurs:

- Vector processing stops or is terminated.

- The sum of the digits collected for the current **`collect digits`** command and the dial-ahead digits exceeds the switch storage limit of 24. Any additional dialed digits are discarded until storage is freed up by a subsequent **`collect digits`** command.

  Note:

  Any asterisk (*) or pound sign (#) digits count towards the 24-digit limit, as do any dial-ahead digits entered after the asterisk or pound sign digit.

- The TTR required by the touch-tone phone user to collect digits is disconnected. This occurs under the following conditions:

  — Successful or unsuccessful **`route-to number`** step is encountered during vector processing except where the number routed to is a VDN extension.

  — Successful or unsuccessful **`route-to digits`** step is encountered during vector processing except where the number routed to is a VDN extension.

  — Successful or unsuccessful **`adjunct routing`** step is encountered during vector processing.

  — Successful or unsuccessful **`converse-on`** step is encountered during vector processing.

  — 10 second timeout occurs, during which time the caller does not dial any digits, asterisks (*) or pound signs (#).

  — A collect ced/cdpd digits step is processed.

  Note:

  When the TTR is disconnected due to a **`route-to number`**, **`route-to digits, converse-on`**, or an **`adjunct routing`** step, all dial-ahead digits are discarded. This means that, following a failed **`route-to`**, **`converse-on`** or **`adjunct routing`** step, a subsequent **`collect digits`** step always requires the caller to enter digits.

  Note:

  Dial-ahead digits are available for use only by subsequent **`collect digits`** commands. The digits are never used by other vector commands that operate on digits (for example, **`route-to digits`**, **`goto...if digits`**, etc.). In addition, these digits are not displayed as part of the CALLR-INFO button operation until they are collected via a **`collect digits`** command.

Collecting CINFO digits – The collect digits step allows you to collect CINFO Digits from the network. When a **`collect ced digits`** or **`collect cdpd digits`** step is processed, the system retrieves the first sixteen ced or cdpd digits from the ISDN User Entered CODE (UEC) Information Element that is associated with the call. It places the digits in the collected digits buffer. Any digits that were in the collected digits buffer when the ced or cdpd digits are collected, are erased. If a TTR was connected to the call from a previous **`collect digits`** step, it is disconnected.

If the ced or cdpd digits contain invalid digits (not 0-9, *, #) the digits are not placed in the collected digits buffer. However, the collected digits buffer is still cleared and if a TTR is attached it is disconnected.

If no ced or cdpd digits were received from the network, when the **`collect ced digits`** or **`collect cdpd digits`** step is reached, the step is skipped. However, the collected digits buffer is still cleared and if a TTR is attached it is disconnected.

A * in the collected digits is treated as a delete character. Only the digits to the right of the * are collected. A # is treated as a terminating character. Only the # and the digits to the left of the # are collected. If a single # is sent, it is placed in the collected digits buffer.

The number of ced or cdpd digits to collect cannot be specified in the **`collect digits`** step. If there are 16 or fewer digits, all the digits are collected. If there are more than 16 digits, the first 16 digits are collected and a vector event is generated.

The CINFO ced and cdpd digits can be used with any vector step that uses the digits in the collected digits buffer.

Once ced or cdpd digits are collected, they can be displayed on a two-line display, or using the callr-info button.

Answer supervision considerations

Answer supervision is provided as soon as a TTR is connected and processing of the command starts. The command always provides answer supervision to an incoming trunk if supervision has not been previously provided except that a collect ced/cdpd digits step does not return answer supervision.

Feature interactions

For BSR and LAI, the command is considered a call acceptance vector command except for collect ced/cdpd digits which is neutral.

CMS/BCMS interactions

Collected digits are passed to the CMS when the **`collect`** step is processed. Digits are not passed to the BCMS.

Consider command

Purpose

The **consider** command defines the resource (split, skill, or location) that is checked as part of a BSR consider series and obtains the data BSR uses to compare resources. After the consider series has been executed, a **queue-to best** or **check best** command can queue the call to the best resource identified.

If the **consider** commands are in a status poll vector, a **reply-best** step returns the data for the best resource found to the primary vector on the origin switch.

Syntax and valid entries

Command	Parameters / Conditions		
consider ____			
	location ____ 1 to 255 (**multi-site BSR only**)[1]		**adjust by** ____ 0-100 percent
	skill _____ 1 to switch max or 1st, 2nd, 3rd VDN skill (**EAS only**)	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	**adjust by** ____ 0- 100 percent

1. This item available with the Virtual Routing feature, only.

Requirements

For switch requirements, see Switch requirements on page 233.

Operation

In order to deliver a call to the resource that can provide the best service, **consider** commands collect and compare information. Whether you use single-site BSR, multi-site BSR, or both, consider steps work very much the same.

Each **consider** command collects status data from one split/skill. Splits or skills on the same switch are identified by number. Remote locations must be identified by a location number assigned on the BSR Application form. See Multi-site BSR applications on page 256 for more information.

Consider commands are typically written in a series of two or more steps called a "consider series." The first step in a consider series collects status data from the resource (a split, skill, or location specified by the user in the command) and saves this data to a buffer. The next **consider** step collects status data on its assigned split/skill and compares the data to that already in the buffer. If the existing data in the buffer indicates the first split/skill can provide better service to the call, the data for the first split/skill remains in the buffer as the "best" data. If the second split/skill can provide better service to the call, its status data replaces the data already in the buffer. Each subsequent step works similarly, collecting data from one resource, comparing it to the "best" data found up to that point, and replacing the best data only if the resource tested by the current step can provide better service to the caller. This series ends when a **queue-to best** or **check-best** step delivers or queues the call, or when a **reply-best** step returns the data for the best resource to a primary vector on the origin switch.

The first consider step in a series shortens the call vectoring 7-step timeout from 1.0 to 0.2 seconds. The timeout is shortened for BSR vectors only (that is, vectors that use **consider** series) in order to reduce real-time delays for call processing and reduce the incidence of race conditions in multi-site BSR applications.

User adjustments

You may have preferences as to which skills should answer certain types of calls. In both single- and multi-site BSR, the **adjust-by** portion of the **consider** command allows you to program these preferences into your vectors.

If a resource does not have an available agent when its **consider** step tests it, the **consider** step collects the Expected Wait Time (EWT) were the call to be queued to that resource. You can adjust this EWT value, for purposes of calculation only, by assigning a value of 0–100 in the user adjustment. The units of this value are supplied by the switch depending on the conditions whenever that **consider** step executes.

For example, in the command **`consider split 1 pri h adjust-by 20,`** the switch interprets **`adjust-by 20`** to mean “add 20% to the EWT, but add at least 20 seconds.” For Expected Wait Times of 1–100 seconds, an adjustment of 20 will therefore add 20 seconds. Above 100 seconds, the same adjustment will add 20% to the EWT for the split/skill specified in
the **`consider`** step.

Important:

If the user adjustment are defined as a number of seconds, BSR would not be efficient when EWT is high. If the user adjustment is defined as a percentage, BSR is not efficient when EWT is low. Such efficiencies become critical in multi-site BSR applications, which involve issues of trunk cost and capacity.

Events that clear “best” data

For DEFINITY software version 9.1 or later, user adjustments also apply to available agent situations (with a strategy other than first found) in a manner that is similar to EWT. For more information, see the “Best Service Routing” section in: *Avaya MultiVantage Call Center Software – Guide to ACD Call Center*, 555-230-716.

As the steps in a consider series execute, the status data for the best resource found is kept in a buffer. This “best” data is unaffected by some call processing events and vector commands, while other events and commands initialize (clear) this buffer. The following table shows you what initializes the best data buffer and what doesn’t.

Initialization of BSR “best” data	
Events and vector commands that clear best data	**Events and vector commands that do not clear best data**
Execution of any queue-to or check command	Converse command

<table>
<tr><th colspan="2">Initialization of BSR “best” data</th></tr>
<tr><th>Events and vector commands that clear best data</th><th>Events and vector commands that do not clear best data</th></tr>
<tr><td rowspan="9">Vector processing terminates:
reply-best command executes
agent answers
successful route-to command
successful adjunct routing command
successful messaging split/skill command
vector disconnect timeout
disconnect command
busy command
vector processing reaches last step without call in queue</td><td>Announcement command</td></tr>
<tr><td>Collect Digits command</td></tr>
<tr><td>Unsuccessful execution of a messaging split/skill command</td></tr>
<tr><td>Unsuccessful adjunct routing command</td></tr>
<tr><td>Goto step/vector with any conditional</td></tr>
<tr><td>Wait command (with any feedback)</td></tr>
<tr><td>Unsuccessful route-to command</td></tr>
<tr><td>Vector processing reaches last step while call is still in queue</td></tr>
<tr><td>Execution of a consider step (this will either replace the current best data with new data or leave the current data untouched)</td></tr>
</table>

Recommendations

It is recommended that you follow the guidelines below when using **`consider`** commands:.

- Don’t put a consider series in vector loops.
- Don’t put any commands between the steps of a consider sequence that would cause a delay. The **`announcement`** and **`wait`** commands, for example, should not be used within a consider sequence. The **`goto`** commands are OK.
- Arrange your **`consider`** steps in order of preference.

  The **`consider`** step that tests the main, or preferred, resource should be the first in the series. The second **`consider`** step should test the resource that is your second preference for handling the given call type, and so on. To avoid unnecessary interflows, put **`consider`** steps for local resources before steps that consider remote resources. Arranging **`consider`** steps in order of preference is recommended for all BSR vectors. It’s especially important when the active VDN for the call is using the 1st-found agent strategy: since the switch will deliver the call to the first available agent found, arranging **`consider`** steps in order of preference will ensure that calls are delivered to the best of the available resources and that unnecessary interflows are avoided.

Answer Supervision Considerations

All forms of the **`consider`** command are ISDN neutral and do not return answer supervision.

Feature interactions

Splits used in **`consider`** commands must be vector-controlled.

CMS/BCMS interactions

BCMS does not log LAI attempts. Therefore, it will not log BSR status polls since they are LAI attempts.

Converse-on command

Purpose

The **converse-on split/skill** command delivers a call to a converse split/skill and activates a voice response script that is housed within a Voice Response Unit (VRU).

Syntax and valid entries

Command **converse-on**	Parameters / Conditions	
skill ____ 1 to switch max 1st, 2nd, 3rd VDN skill (**EAS only**)	**pri ___** **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	**passing ____ and ____** 6-digit string, "*" , "#" , "none"[1], "ani", "vdn", "digits", "qpos", "wait"
split ____ 1 to switch max	**pri ___** **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	**passing ____ and ____** 6-digit string, "*" , "#" , "none"[1], "ani", "vdn", "digits", "qpos", "wait"

1. If the first "passing" value is entered as "none" the second value must also be entered as "none."

Requirements

A converse split must be vector-controlled.

Operation

The **converse-on** command is designed primarily to integrate Voice Response Units (VRUs), principally the Conversant Voice Response Unit (VRU), with the switch. The command effects data passing between the switch and the VRU, and it enables the caller to hear the appropriate voice response script housed in the VRU.

For details regarding call flows, data passing, collection, and return specifications involving the **converse-on** command, see Appendix J: Call flow and specifications for converse – VRI calls on page 585.

If the command is successful, it delivers the call to a predetermined split/skill, which is referred to as the converse split/skill. Once the call is answered by the VRU, the command may or may not pass data to the VRU (depending upon the parameters of the command). Regardless of whether or not data is passed, the caller is then connected to the VRU, which in turn executes the voice response script. If by this time the call has already queued to a nonconverse split/skill, the call retains its position in the nonconverse split/skill queue. If an agent from the nonconverse split/skill becomes available to service the call while the voice response script is being executed, the switch drops the line to the VRU and connects the caller to the available agent. The VRU, in turn, detects the disconnect and terminates the voice response script. Whenever a voice response script is executed, any audible feedback provided by the vector is disconnected, and no further vector steps are executed until the voice response script is executed.

The VRU may or may not eventually return data to the switch. If, once the voice response script is completed, there is no data to be returned from the VRU to the switch, the VRU drops the line to the switch, and vector processing is reactivated on the switch.

If there is data to be returned to the switch, the Converse data return code is outpulsed before the data to be passed is outpulsed. Once all VRU data is received, it is stored in the Call Prompting digits buffer as dial-ahead digits, and vector processing is reactivated. Digits returned by the VRU are not heard by the caller.

Digits returned from the VRU can be:

- Displayed on the answering agent's display set (automatically for 2-line displays, or by using the **CALLR-INFO** button for 1-line displays)
- Treated as an extension in a **`route-to digits`** step
- Used for vector conditional branching in a step containing a command with the **`if digits`** parameter
- Tandemed to an ASAI host

The switch can be set up to pass information in-band to the VRU. In such a case, the **`converse-on`** command can outpulse up to two groups of digits to the VRU. The digits may serve two major purposes: the digits may notify the VRU of the application to be executed, and they may share call related data, such as ANI (BN) or caller digits collected by the switch. (In many applications, both application selection and data sharing are required.) The touch tone outpulsing rate is adjustable see Appendix J: Call flow and specifications for converse – VRI calls on page 585 for details.

Since in many cases the digit strings are of variable length, the switch always appends a pound sign (#) character to the end of each digit string. The **`Prompt`** and **`collect`** steps in the voice response script must therefore always be administered to expect # as the end-of-string symbol and to include # in the digit count.

The sending of # prevents excessive delays caused by digit timeouts, and it prevents other problems caused by timeouts. It also ensures that each data field is used to satisfy a single **`prompt`** and **`collect`** step.

Any data passed from the switch to a VRU is outpulsed in-band. The user can administer two time delays on the System Parameter Features form: converse first data delay and converse second data delay fields. These delays may range from 0 to 9 seconds with a default of zero seconds for the converse first data delay and a default of two seconds for the converse second data delay. The delays are needed to give the VRU time to invoke an application and to allocate a touch-tone receiver to receive the passed digits.

Note:

No time delays are invoked when the keyword **none** is administered.

If <data_1> is not **none**, the converse first data delay timer starts when the call is answered by the VRU. When the timer expires, the <data_1> digits are outpulsed in-band to the VRU. The end-of-string character (#) is then outpulsed.

If <data_2> is not **none**, the converse second data delay timer starts when the end-of-string character (#) from the first digit string is outpulsed. When the timer expires, the <data_2> digits are outpulsed in-band to the VRU. The end-of-string character (#) for the second digit string is then outpulsed.

The following values may be administered for <data_1> and <data_2> within the **`converse-on`** command:

- **Administered digit string**: This string can contain up to six characters consisting of one or more digits (0 through 9) or asterisks (*). The pound sign (#) may not be included in a digit string because it is reserved as the end-of-string character. However, a single **#** may be administered.
- **ani**: If the call is an internal call or an incoming DCS call, this data type causes the extension of the calling party to be outpulsed. If the call is an incoming ISDN-PRI or R2MFC Signaling call with ANI (BN) provided to the switch, the calling party number/billing number (CPN/BN) of the calling party is outpulsed to the VRU. If there is no ANI (BN) to send, the end-of-string pound sign (#) is the only character outpulsed. Any other type of incoming call results in **#** being outpulsed.
- **digits**: This data type can be used only if Call Prompting is optioned. To pass CINFO digits, Vectoring (CINFO) must also be enabled. The digits data type causes the most recent set of digits collected in vector processing, either from the caller or from the network, to be outpulsed. If no digits are available, the end-of-string pound sign (#) is the only character outpulsed.
- **none**: This data type causes no characters to be outpulsed. Also, no end-of-string pound character (#) is outpulsed, and no time delays are invoked.
- **qpos**: This data type causes the value of the queue position of a call in a nonconverse split to be outpulsed. This value is a variable length data item from which between one and three digits can be outpulsed. If the call is not queued, the end-of-string pound sign (#) is the only character that is outpulsed. This data may be used by the VRU to inform callers of their position in queue or to decide whether to execute a long or short version of a voice response script.

Note:

The use of this keyword is not recommended with multiple split/skill queuing. Any queue position value that is sent may not be meaningful. If the call is queued to multiple nonconverse splits/skills, the value of the caller's queue position in the first nonconverse split/skill is sent. Priority queuing (priority assigned to the queue vector step) and Dynamic Queue Position, which is available with Avaya Business Advocate, can put subsequent calls into the queue ahead of the waiting call.

- **vdn**: This data type causes the VDN extension to be outpulsed. In cases where multiple VDNs are accessed, normal VDN override rules determine which VDN extension is outpulsed.
- **wait:** This data type can be used only if the Vectoring (G3V4 Advanced Routing) customer option is enabled. It causes the expected wait time of the call in seconds to be outpulsed. See Expected Wait Time (EWT) on page 126 for a detailed description of expected wait time. If the call is not queued or if it is queued only to splits that are unstaffed or splits where all agents are in AUX work mode, the end-of-string character # is the only character outpulsed. The value outpulsed is a variable number not padded with zeroes. It is a maximum of four digits always followed by #. The range is 0# to 9999# or a single #.
- **#**: This is the only character outpulsed. Outpulsing this character causes the corresponding **`prompt`** and **`collect`** command in the voice response script to be skipped.

A pound character (#) is always outpulsed at the end of each digit string. Where **#** is administered, or where the **digits** keyword is administered and the last digit collected from the caller is **#**, only one **#** is outpulsed. No **#** is outpulsed when the keyword **none** is administered.

If **`data_1`** is administered as **none**, **`data_2`** must also be **none**.

Answer supervision considerations

Answer supervision is returned only once during the life of a call. If a call is answered as a result of a **`converse-on`** step, answer supervision is sent only if it has not been sent previously. If digits are passed to the VRU, answer supervision is not sent until after the digits are outpulsed.

Feature interactions

Abandon Call Search – If the **`converse-on`** step places a call to a hunt group, and if the incoming call was placed via a trunk group with Abandon Call Search activated, the system checks that the calling party has not abandoned the call (that is, hung up) before terminating to an agent.

Adjunct Switch Applications Interface (ASAI) – Since vector-controlled splits/skills cannot be ASAI-monitored domains, ASAI cannot be used to supplement the operation of the **`converse-on`** step.

If a **`converse-on`** step places a call to an ASAI-monitored domain, ASAI event messages are sent over the ASAI link.

Whenever a **`converse-on`** step places an ASAI-monitored call, the ALERTing message sent to the ASAI host includes a Cause IE, Coding Standard 3 value 23 (CS3/23). This informs the ASAI host that the call has not been de-queued from any nonconverse splits/skills.

If a **`converse-on`** step is executed while an adjunct routing request is outstanding, the route request is canceled.

Audio Information Exchange (AUDIX) – If a **`converse-on`** step calls the AUDIX, the call is treated as a direct call to the AUDIX. The caller hears the “welcome to AUDIX” message and may retrieve his or her messages in the usual manner.

If a call is forwarded to or covers to a VDN and is then delivered to an AUDIX hunt group by a **`converse-on`** step, the call to the AUDIX is treated as a redirected call, and the caller may leave a message for the principal.

Auto-Available Splits/Skills – A **`converse-on`** step may place a call to an auto-available split/skill. Except in cases where the converse split/skill is ASAI-controlled, auto-available converse splits/skills are recommended for Voice Response Integration (VRI).

Call Coverage – Call Coverage does not apply because the **`converse-on`** step may deliver calls only to vector-controlled splits/skills, which do not have coverage paths.

Call Detail Recording – For incoming calls to a VDN, the duration of the call is recorded from the time answer supervision is returned. Answer supervision is returned for a successful **`converse-on`** step. No ineffective call attempt records are generated for **`converse-on`** steps that fail. Also, no outgoing calls can be placed by a **`converse-on`** step.

Call Park – Calls placed by a **`converse-on`** step may not be parked.

Call Pickup – Calls placed by a **`converse-on`** step ringing at an agent station may be picked up if that agent is part of a pickup group. Subsequent transfers are denied.

Call Prompting – The Call Prompting customer option must also be enabled to gain full VRI functionality. Without Call Prompting, any data returned by the VRU cannot be collected and processed by the switch.

If the **`converse-on`** step places a call to a split/skill of live agents, any digits collected previously may be displayed by agents using the callr-info button.

Call Vectoring—Basic – The **`converse-on`** step is an enhancement to the Basic Call Vectoring customer option. This option must be enabled in order to invoke the VRI feature.

Class of Restriction (COR) – As is the case for the **`queue-to split/skill`** and **`check split/skill`** vector steps, no COR checking is carried out when a **`converse-on`** step places a call to a split/skill.

Conference – Any attempt to conference a call placed by a **`converse-on`** step is denied.

Coverage Callback – A call placed by a **`converse-on`** step does not follow any coverage paths. Therefore, Coverage Callback is not available. Also, if a call reaches a **`converse-on`** step via a VDN in a coverage path, coverage callback cannot be used.

Direct Department Calling (DDC) – A converse split may be administered as a direct department calling split.

Distributed Communications System (DCS) – If an incoming DCS call is placed to a vector with a **`converse-on split/skill x pri y passing ani ...`** step, the DCS extension of the calling party is outpulsed.

Priority Levels – A call placed by a **`converse-on`** step may be queued at one of four priority levels: low, medium, high or top.

Hunt Groups – The **`converse-on`** step may deliver a call to a vector-controlled hunt group, ACD split/skill, Message Center or an AUDIX hunt group.

Integrated Services Digital Network (ISDN) – The **`converse-on`** step may be administered to outpulse to the VRU the ANI (calling party number/billing number CPN/BN) of the calling party via use of the **ani** keyword.

Intercept Treatment – A caller is never given intercept treatment upon execution of a **`converse-on`** step. Failing to place a converse call successfully results in the failure of the **`converse-on`** step. Vector processing continues at the next vector step.

Interflow – Since a **`converse-on`** step can place calls only to hunt groups that are vector-controlled, and since the activation of Call Forwarding for a vector-controlled hunt group is blocked, calls placed by a **`converse-on`** step to a hunt group cannot interflow.

Intraflow – Since a **`converse-on`** step can place calls only to hunt groups that are vector-controlled (that is, without coverage paths), intraflow is not possible.

Live Agents – Although not recommended, the switch does not prevent a **`converse-on`** step from delivering a call to a group of live agents. To the agent, the call looks like any other ACD call. However, certain features, such as call transfer, conference, and supervisor assist are denied.

The answering agent can display any digits collected prior to executing the **`converse-on`** step by using the **callr-info** button.

Look-Ahead Interflow (LAI) – If a call placed by a **`converse-on`** vector step is answered by a VRU, or if such a call queues to a split/skill on the receiving switch while a LAI call attempt is outstanding, the LAI call attempt is accepted.

A **`converse-on`** step that fails is neutral.

Message Center – The **`converse-on`** step may deliver calls to message hunt groups. Such calls are treated as direct calls to the message.

If a call is forwarded to a VDN and then delivered to a message split by a **`converse-on`** step, the call is treated as a redirected call.

Multiple Split/Skill Queuing – A call can be queued to three different splits/skills and then to a converse split/skill as a result of a **`converse-on`** step.

Music on Hold – During the data return phase of a **`converse-on`** step, the caller is temporarily placed on hold. Music on hold, if administered, is suppressed.

Non-Vector Controlled Splits/Skills – A **`converse-on`** step may not place a call to a nonvector-controlled split/skill.

Priority Queuing – The queue priority of a call placed by a **`converse-on`** step is administrable on the vector step.

Queue Status – All queue status display, queue status indication and queue warning wall lamp feature capabilities also apply to calls queued by the **`converse-on`** command.

Queuing – Calls handled by the **`converse-on`** step queue when they are delivered to busy hunt groups. Call Vectoring audible feedback is not disconnected while a converse call is in queue.

If a **`converse-on`** step is executed while a call is queued to a nonconverse split/skill, the call remains in queue for the nonconverse split/skill.

The queue priority of the call is administrable on the vector step.

Recorded Announcement – VRI may be used to increase the system's recorded announcement capacity by off-loading some recorded announcements to the VRU. Callers can be redirected by the **`converse-on`** step to a group of VRU ports and use data passing to specify the correct announcement to play.

Redirection on No Answer (RONA) – If a **`converse-on`** step places a call to a hunt group with a **no answer timeout** administered, and if the call rings at an agent terminal/port for longer than the administered timeout, the call is redirected, and the agent/port is put into the AUX work state (or logged out if the agent is a member of an auto-available split/skill).

Thereafter, under RONA, the call is requeued to the split/skill unless there is no room in the queue or unless this is an auto-available split/skill whose agents are all logged out. If the call cannot be requeued, the **`converse-on`** step fails, a vector event is logged, and vector processing is restarted at the next vector step.

Service Observing – Calls placed by a **`converse-on`** step may be service observed. To prevent the observer from hearing tones being outpulsed to the VRU, the observer is not connected to the call until the data passing phase is complete. If data is returned by the VRU, the observer is put in service observing pending mode, and the calling party is temporarily put on hold while the VRU digits are outpulsed. Upon completion of the converse session, and once the VRU hangs up the line, the observer remains in service observing pending mode.

It is not recommended that a service observing warning tone be administered since the warning tone may interfere with the interaction between the VRU and the calling party.

System Access Terminal (SAT) – `converse-on` steps may be administered from the SAT terminal.

System Measurements – System measurements track converse calls to hunt groups and attendant groups.

Timed After Call Work (ACW) – Timed ACW cannot be assigned to auto-available splits (AAS). If a call to a VDN with Timed ACW routes to a converse split, the VDN Timed ACW does not apply.

If Timed ACW is assigned to a non-AAS split that is a converse split, the Timed ACW of the split does apply.

Touch-Tone Dialing – Any touch-tone dialing by the calling party during the digit passing phases of a session involving a **`converse-on`** step does not result in corruption of data or in the collection of this data in the form of dial-ahead digits by the switch.

Only after the digit passing phase from the switch to the VRU is completed can the calling party enter touch-tone digits in response to a VRU prompt. Only after the VRU to the switch data return phase is completed and an additional **`collect digits`** vector step is executed can the calling party enter a touch-tone response to a switch prompt.

Transfer – A call placed by a **`converse-on`** step may not be transferred. The only form of transfer allowed is the data passing operation during the data return phase at the end of a voice response script.

If an illegal attempt to transfer a converse call is made, a vector event is logged, the line to the VRU is dropped, and vector processing is reactivated at the next vector step.

If an illegal transfer is attempted by a live agent with a multifunction set, the transfer is denied and the agent may reconnect to the call.

Transfer out of AUDIX – If a **converse-on** step delivers a call to an AUDIX hunt group, and if the calling party then attempts to transfer out of AUDIX, the transfer fails, and vector processing is reactivated at the next vector step.

Uniform Call Distribution (UCD) – A converse split/skill may be administered as a Uniform Call Distribution split/skill.

VDN as a Coverage Point – If a call covering to a VDN is processed by the **converse-on** command and subsequently reaches a station user (that is, a member of a converse split/skill), and if the converse split/skill agent attempts to activate Consult (coverage), or Coverage Leave Word Calling, any of these coverage attempts is denied because the call is still in vector processing. If the converse split/skill is an AUDIX/Message Center split/skill, the call covered to the VDN is treated like a redirected call to the AUDIX/MCS; the original principal and reason for redirection is used in the same manner as a Call Forwarded call to a VDN.

VDN Override – If a call that accesses multiple VDNs encounters a **converse-on** step passing **vdn**, normal override rules determine which VDN number is outpulsed to the VRU.

VDN Reports – For call tracking in the CMS and BCMS VDN reports, a **converse-on** step is treated like an **announcement** step. A call is considered “answered” when it is answered by a nonconverse split/skill but never when it is answered by a converse split/skill.

Vector-controlled Splits/Skills – A **converse-on** step may place a call to a split/skill only if that split/skill is administered as a vector-controlled split/skill.

CMS interactions

The CMS tracks calls placed by a **converse-on** step to a CMS-measured split/skill. Since a **converse-on** step allows a call to be “answered” in more than one split/skill, trunk totals no longer match split/skill totals. However, VDN totals and trunk totals will match.

For call tracking in the CMS VDN reports, a **converse-on** step is treated like an **announcement** step. A call is considered answered when it is answered by a nonconverse split/skill but never when it is answered by a converse split/skill.

BCMS interactions

BCMS tracks calls placed by a **converse-on** step to a BCMS-measured split/skill. Since a **converse-on** step allows a call to be “answered” in more than one split/skill, trunk totals no longer match split/skill totals. However, VDN totals and trunk totals will match.

For call tracking in BCMS VDN reports, a **converse-on** step is treated like an **announcement** step. A call is considered answered when it is answered by a nonconverse split/skill but never when it is answered by a converse split/skill.

Disconnect command

Purpose

The **`disconnect`** command ends treatment of a call and removes the call from the switch. Also allows the optional assignment of an announcement that will play immediately before the disconnect.

Important:
You should always warn the caller prior to disconnecting the call.

Syntax and valid entries

Command	Parameters / Options
`disconnect`	**`after announcement _____`** `extension no. or “none”`

Requirements

The relevant announcements must be administered and recorded.

Operation

While the command’s optional announcement is playing, the call remains in queue and can be connected to an agent. When the announcement completes (or is not specified), the command forces a disconnect, ends the treatment of the call, and removes the call from the switch.

Answer supervision considerations

If the switch has not yet sent answer supervision, the switch does so immediately before disconnecting the call, whether an announcement is specified or not. If an announcement is specified, answer supervision is given before an attempt is made to connect the announcement. The exception is for ISDN calls, where the disconnect can occur without answer supervision being sent when an announcement is not played.

Feature interactions

For LAI, the command can be considered either a call acceptance vector command or a call denial vector command.

The command is considered a call acceptance vector command whenever an announcement is included within the command and one of the following is true:

- Announcement is available.
- Call is queued for an announcement.
- Announcement is retried.

The command is considered a call denial vector command whenever one of the following is true:

- No announcement is included within the command.
- Announcement is included within the command, but the announcement is unavailable.

CMS interactions:

Disconnect command

Database Item	Report Heading
DISCCALLS/DISCTIME	Calls Forced Disc
	Calls Busy/Disc
OTHERCALLS/OTHERTIME	Inbound Other Calls
INTIME	Avg Time In Vector

DISCTIME, OTHERTIME, and INTIME for splits and vectors are tracked according to when the announcement starts. DISCTIME, OTHERTIME and INTIME for VDNs are tracked according to when the trunk idles.

BCMS interactions

A call that is disconnected via the command is tracked as OTHER in the VDN Report.

Goto step and goto vector commands

Purpose

The **`goto step`** command allows conditional or unconditional movement (branching) to a preceding or subsequent step in the vector.

The **`goto vector`** command allows conditional or unconditional movement (branching) to another vector. The goto vector step does not remove a call from queues in which it is already placed.

All parameters, options and value limits are identical for the **`goto step`** and **`goto vector`** commands.

Syntax and valid entries

Command	Parameters / Conditions			
goto **step** _____ 1-32 or **vector** ______ 1 to switch max				
if				
ani ___ <,>,<=,>=,<>,=	_____ 1-16 characters including digits (0-9), “?”, “+” “none”[1]			
in, not-in	**table** ____ 1 100[2]			
available-agents				
in skill ____ 1 to switch max, or 1st, 2nd, or 3rd for VDN Skill			____ <,<= >,>=,<>,=	____ 1 to 1500[2] 0 to 1499[2]
in split _____ 1 to switch max			____ <,<= >,>=,<>,=	____ 1 to 1500 0 to 1499[2]
calls-queued				
in skill ____ 1 to switch max, or 1st, 2nd, or 3rd for VDN Skill (**EAS-only**)	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority		____ <,<= >,>=,<>,=	____ 1 to 999[2] 0 to 998[2]
in split ____ 1 to switch max	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority		____ <,<= >,>=,<>,=	____ 1 to 999[2] 0 to 998[2]
counted-calls	**to vdn** ____ vdn extension, ”latest” or “active”[3]	____ <,<= >,>=,<>,=	____ 1 to 999[2] 0 to998[2]	
digits ___	_____			
<,>,<=,>=,<>,=	1-16 digits, including 0-9, “#”, “?”, “+”, or “none”[1]			
=	meet-me-access[4]			
in, not-in	**table** ____ 1 100[2]			

Command	Parameters / Conditions				
goto step (or vector) (*continued*)					
if (*continued*)					
expected-wait					
for ___ best, call	____ <,>,<=,>=,<>,=		____ 0-9999 seconds		
for ___ split	____ 1-switch max	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	____ <,>,<=,>= <>, =	____ 0 to 9999 seconds	
for ___ skill (**EAS-only**)	____ 1-switch max, 1st, 2nd, 3rd skill for VDN	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	____ <,>,<=,>= <>, =	____ 0 to 9999 seconds	
holiday ____ in, not-in	**table** ____ 1-10				
ii-digits	____ <,>,<=,>= <>, =	____ 2-digit string, "+", "?", or "none"[1]			
	in, not-in	**table** ____ 1 to 100[2]			
interflow-qpos	____ <,>,<=,>= <>, =	____ 1-9			
meet-me-full (go-to step, only)					
meet-me-idle (go-to step, only)					
no match[5]					
oldest-call-wait					
	in skill _____ 1 to switch max, or 1st, 2nd, 3rd, skill for VDN (**EAS-only**)	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	____ <,>,<=,>= <>, =	____ 0 to 999 seconds	
	in split ____ 1 to switch max	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	____ <,>,<=,>=, <>,=	____ 0 to 999 seconds	
queue-fail[6]					

Command	Parameters / Conditions			
goto step (or vector) (*continued*)				
if (*continued*)				
rolling-asa	**for** ____			
	skill (**EAS-only**)	______ 1 to switch max, or 1st, 2nd, 3rd skill for vdn	____ <,>,<=,>= <>, =	____ 0-999 seconds
	split	______ 1 to switch max	____ <,>,<=,>= <>, =	____ 0-999 seconds
	vdn	______ vdn extension, “latest” or “active”[3]	____ <,>,<=,>= <>, =	____ 0-999 seconds
staffed-agents				
		in skill ______ 1 to switch max, or 1st, 2nd, 3rd skill for vdn	____ <,>,<=,>=, <>,=	____ 1-1500[2]
		in split ______ 1 to switch max	____ <,>,<=,>=, <>,=	____ 1-1500[2]
time-of-day	**is**	____ ____ ____ mon-sun, “all” 00-23 hour 00-59 minute	**to**	____ ____ ____ mon-sun, “all” 00-23 hour 00-59 minute
unconditionally				
wait-improved for	______			
	best	________ <,>,<=,>= <>, =		______ 0 to 9999 seconds
	skill ____ 1 to switch max, or 1st, 2nd, 3rd skill for vdn (**EAS-only**)	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	______ <,>,<=,>=, <>,=	____ 0-9999 seconds
	split ____ 1 to switch max	**pri** ____ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	____ <,>,<=,>=, <>,=	____ 0-9999 seconds

1. The question mark (?) is a wild card that matches any digit (0-9) at the specified position. The plus sign (+) matches any or no characters at the specified position.
2. Maximum limit may be less on some platforms. Use the help key for your switch administration software to determine the applicable limits for your system.

3. “Active” refers to the VDN specified by VDN Override settings. “Latest” refers to the VDN specified for the current vector.
4. This item available with **`meet-me conference`** vectors, only.
5. This item available with **Dial by Name** feature, only.
6. This item available with **Attendant Vectoring** feature, only.

Requirements

For more information about options required to enable the **`goto`** commands, see MultiVantage options required to enable vector commands on page 389.

Operation

If the command syntax includes **`unconditionally`**, the command always branches. The unconditional form of the command is commonly used for skipping vector commands as well as for looping through vector commands.

Otherwise, branching takes place according to one of the conditions that follow:

- The average speed of answer for the indicated split/skill or VDN meets the constraints defined by the comparator and threshold value.
- The number of available agents in the indicated split/skill meets the constraints defined by the comparator and the threshold value.
- The number of queued calls in the indicated split/skill and at the specified priority level (or higher) meets the constraints defined by the comparator and the threshold value.
- The number of active calls in the indicated VDN meets the constraints defined by the comparator and the threshold value.
- The expected wait time at the specified priority level for the indicated split/skill, or for the call meets the constraints defined by the comparator and the threshold value.
- The oldest call-waiting in the indicated split/skill at the specified priority level (or higher) has been waiting for a period of time within the constraints defined by the comparator and the threshold value, which is expressed in seconds.
- The number of staffed agents in the indicated split/skill meets the constraints defined by the comparator and the threshold value.
- Digits collected via the **`collect digits`** command match the criteria defined by the comparator for the specified digit string. Or, the digits are found or not found, depending upon the option chosen, in the specified Vector Routing Table. The # digit can be tested against as a single digit.
- The ani digits match the criteria defined by the comparator for the specified digit string. Or, the ani digits are found or not found, depending upon the option chosen, in the specified Vector Routing Table.

- The II-digits match the criteria defined by the comparator for the specified digit string. Or, the II-digits are found or not found, depending upon the option chosen, in the specified Vector Routing Table.
- Time-of-day criteria are met.

  Note:

  The syntax for this condition can be illustrated by a couple of examples, as follows: **`mon 8:01 to fri 17:00`** means anytime between 8:01 A.M. Monday through 5:00 P.M. Friday, and **`all 17:00 to all 8:00`** means between 5:00 P.M. and 8:00 A.M. on any day of the week.
- The Expected Wait Time (EWT) for the call is decreased by a period of time within the constraints defined by the comparator and the threshold value, which is expressed in seconds. The improvement in EWT is defined by calculating the difference between the call's current EWT and its EWT were it to be queued to the resource specified in the command.
- The call's position in the interflow-eligible portion of the queue meets the condition defined by the comparator and the threshold value (representing queue position counting backward from 1, which is the head of the eligible queue).
- For Attendant Vectoring, there is no way to check ahead of time to see if a call can queue, and there is no way to check if, after the fact, a call queued successfully. The **`queue-fail`** command allows you to provide additional routing if a call to an attendant vector fails. You can redirect the call to another step or to another vector if the call cannot be queued.

Answer supervision considerations

The call answer is not affected by the command.

Feature interactions

For BSR and LAI, the command is considered a neutral vector command in all cases. When a call experiences Look Ahead interflow, the ANI value is sent along with the call only for ISDN PRI calls. ANI is not sent for internal or DCS calls.

CMS/BCMS interactions

The **`goto step`** command is not tracked on the CMS or on the BCMS.

The ANI and/or II-digits are passed to the CMS when the call first starts vector processing if the following is true:

- Basic Call Vectoring and/or Call Prompting is optioned
- ANI is available from the network, the call is internal, or is received over DCS

- II-digits is available from the network
- The CMS is R3 (R3V5 for II-digits) or a newer version

ANI and II-digits are not passed to BCMS.

The **`goto vector`** command is tracked on CMS. The following database items are created.

goto Vector command		
Database Item	**Report Heading**	**Notes**
OUTFLOWCALLS/ OUTFLOWTIME GOTOCALLS/ GOTOTIME	Vector Flow Out	
INTIME	Avg Time In Vector	
INFLOWCALLS	Vector Flow In	new vector

CMS interaction notes for `goto vector`: – The ANI and/or II-digits is passed to the CMS when the call first starts vector processing if the following is true:

- Basic Call Vectoring and/or Call Prompting is optioned
- ANI is available from the network, the call is internal, or is received over DCS
- II-digits is available from the network

ANI and II-digits are not passed to BCMS.

Messaging command

Purpose

The **messaging split/skill** command allows the caller to leave a message for the specified extension or the active or latest VDN extension (default).

Syntax and valid entries

Command	Parameters / Conditions	
messaging		
	skill ____ 1 to switch max, or 1st, 2nd, 3rd skill for vdn (**EAS-only**)	**for extension ____** extension no., “latest” or “active”[1]
	split ____ 1-switch max	**for extension ____** extension no., “latest” or “active”[1]

1. “Active” refers to the VDN specified by VDN Override settings. “Latest” refers to the VDN specified for the current vector.

Requirements

The split/skill involved must be an AUDIX split/skill, a remote AUDIX split or skill (DCS-AUDIX).

Operation

This command causes the caller to be connected to the AUDIX or Message Center split/skill so that the caller may leave a message for the specified extension (call answering service or “mail”).

If the split/skill number specified in the command is a valid message service split/skill (such as an AUDIX), and if the extension is either a valid assigned extension or is administered as active or latest the system attempts to terminate the call to the message service split/skill for call answering service.

If the call is queued to the message service split/skill, or if the call terminates to an available message service agent or an AUDIX voice port, the caller is connected to ringback (signifying successful termination), and vector processing terminates. Termination is unsuccessful, and vector processing continues at the next vector step if any one of the following is true:

- Split/skill queue is full.
- AUDIX link is down.
- All AUDIX voice ports are out of service.
- Message service split/skill is DCS-AUDIX and all DCS trunks are busy.

If call termination is successful, and if the administered extension (or default VDN) is a message service subscriber, the caller can leave a message for the specified extension.

Note:

Agent and/or supervisor stations may be equipped with Automatic Message Wait (AMW) lamps to accommodate the "mail" specified in the **`messaging split/skill`** command. The lamps can be assigned for VDNs or extensions used to access the messaging split/skill and for which messages are to be left. When messages are left for these VDNs or extensions, the assigned AMW lamps light.

If the extension or VDN is not a subscriber of the message service, the caller receives ringback until he or she disconnects.

Answer supervision considerations

If answer supervision has not already been returned, it is returned when the messaging service port or station is connected to the call (that is, when the call is answered by the port or station).

Feature interactions

The command can use an AUDIX hunt group in its operation.

If the command specifies a specific "mailbox" extension, the original principal for a call covered by a VDN is not passed to the adjunct, and it does not appear in the display to the answering agent. The specified extension appears in the display.

If the command is accessed via a direct call to the VDN, and if the mailbox is administered as **active** or **latest**, the corresponding active or latest VDN extension mailbox is sent to the messaging adjunct. Additionally, if the call is sent to a switch Message Service split/skill, the associated VDN name is sent to the messaging adjunct.

If the command specifies active or latest as the mailbox extension, the original principal for a call covered to or forwarded to a VDN is used as the default mailbox for the call instead of the active or latest VDN. Accordingly, the original principal extension and the reason for redirection are passed to the messaging adjunct, and they subsequently appear in the display to the answering agent.

AUDIX does not support mixed length numbering plans.

If the command leaves a message for a VDN or for another messaging service extension, the Automatic Message Waiting Lamp (AMWL) associated with the VDN or extension lights steady.

For LAI, the command can be considered as either a call acceptance vector command or a neutral vector command.

The command is considered a call acceptance vector command whenever one of the following is true:

- Call terminates to an agent or to an AUDIX port.
- Call queues to a messaging split/skill.

The command is considered a neutral vector command whenever the command fails.

CMS interactions

When a queued call successfully goes to the messaging split, OUTFLOWCALLS/OUTFLOWTIME (1st split/skill) and DEQUECALLS/DEQUETIME (2nd/3rd splits [skills]) are tracked in the split/skill tables. These calls are reported as split/skill Flow Out, Dequeued Calls, and Dequeued Avg Queue Time.

Calls that queue via a **`messaging split/skill`** command are tracked as CALLSOFFERRED and LOWCALLS (no priority) or MEDCALLS (priority). These calls are shown in the standard reports according to the final disposition of the call.

The presence of the command in a vector enables the calls serviced by the vector to be vector-directed. When such a call is answered by an agent, the call is tracked as ACDCALLS/ANSTIME, and it is reported as ACD Calls, Split/Skill ACD Calls, and Avg Speed Ans.

Finally, if the command directs a call to a split/skill, the BACKUPCALLS database item is incremented, and the call is reported as Calls Ans in Backup and Calls Handled/Backup. The Calls Ans in Main report item is calculated by using the algorithm ACDCALLS - BACKUPCALLS.

A call abandoned after the command routes the call to a station or to an attendant is tracked as ABNCALLS/ABNTIME for the messaging split/skill and in the VDN/vector tables.

BCMS interactions

A call advanced to another position via the command is tracked as an outflow in the VDN Report.

Queue-to command

Purpose

The **queue-to** command unconditionally queues a call to a split/skill, attendant group, attendant, or hunt group, and assigns a queuing priority level to the call in case all agents or attendants are busy.

Syntax and valid entries

Command	Parameters / Conditions	
queue-to		
	attd-group[1]	
	attendant[1] ________ extension no.	
	best	
	hunt-group[1] _____ 1 to switch max	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority
	skill ____ 1 to switch max, or 1st, 2ns, 3rd skill for vdn (**EAS-only**)	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority
	split ____ 1 to switch max	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority

1. This item available with Attendant Vectoring feature, only.

Requirements

The split/skill involved must be vector-controlled.

Operation

A call sent with this command either connects to an available agent or attendant in the specified resource or enter the resource's queue. When it enters the queue, feedback is not given to the caller by this command.

Note:

In Attendant Vectoring, a **wait-time 0 secs hearing ringback** step should be used to give immediate feedback to the caller. The **queue-to** command does not provide ringback until the call is actually ringing the attendant. The **wait-time** step should be implemented as the first vector step or as the step immediately before the **queue-to** step.

If single-site BSR is enabled, **`queue-to best`** queues or delivers a call to the best local split/skill found by a consider series. If multi-site BSR is enabled, the best resource may be at a remote location; in this case, **`queue-to best`** interflows the call to the interflow VDN defined for that location on the BSR Application form.

A call may be queued to up to three local split/skill simultaneously. A call remains queued either until vector processing terminates (via a **`disconnect`**, **`busy`**, or **`route-to`** command, or via a dropped or abandoned call) or until the call reaches an agent. When an agent becomes available in any split/skill to which the call is queued, the following actions take place:

- Call begins ringing the agent.
- Call is removed from any other queues.
- Vector processing terminates.

If the entered split/skill is one of the split/skill to which the call is already queued, the call is requeued at the new priority level. If the priority level specified is the same as the priority level at which the call is queued, the call remains in the same position in queue. The step is skipped, and vector processing continues at the next step if any of the following conditions are true:

- Desired split/skill's queue is full.
- Desired split/skill's is not vector-controlled.
- Desired split/skill's has no queue and also no available agents.
- Call has been previously queued to three different split/skills.

Note:

A **`route-to`** to another VDN can be used to remove the call from the splits it is queued to if necessary. The steps in the routed-to vector then can be used to queue to other splits.

A **queue-to best** command will have the same operation and interactions as the **queue-to split/skill** command when the best resource is a local split/skill. When the best resource is at a remote location, the **queue-to best** command will function as an unconditional **route-to** command (with cov=n) performing LAI.

When a **queue-to best** command executes, it initializes the data for the best resource (the “best” data) the consider series found for this call. If no “best” data has been defined by the consider series, a vector event is logged and processing continues at the next vector step. A consider series might not produce “best” data for any of the following reasons:

- all resources considered are unstaffed
- no resource considered has an open queue slot
- “best” data has been initialized before execution of the **reply-best** step (because there are no consider steps in the status poll vector or because the vector contains a prior step that initializes “best” data).

For a list of events and vector commands that initialize “best” data produced by consider series, see Events that clear “best” data on page 425.

If a queue attempt to a local resource fails, a vector event is logged and processing continues at the next vector step. The “best” data is initialized.

If an interflow attempt to a remote resource fails, a vector event is logged and processing continues at the next vector step. If a local split/skill was identified as best at some point in the consider series before the interflow attempt, the call is queued to the local resource. Whether or not the call can be queued locally in this case, the “best” data is initialized and processing continues at the next vector step.

Answer supervision considerations

Answer supervision is returned (if not already returned) when the call is connected to an answering agent.

Feature interactions

The **queue-to** command can access an AUDIX split/skill in cases where a VDN is assigned as a coverage point. To enable this function, the split/skill must be assigned as a vector-controlled hunt group.

For BSR and LAI, the command can be considered either a call acceptance vector command or a neutral vector command.

The command is considered a call acceptance vector command whenever one of the following is true:

- Call terminates to an agent.
- Call queues to a split/skill.
- BSR interflowed call is accepted at remote interflow vector.

The command is considered a neutral vector command when the call neither terminates nor queues.

No COR checking is carried out when a queue-to step places a call to a split/skill.

CMS interactions

Calls queued via a **`queue-to split/skill`** command are tracked as CALLSOFFERRED and LOWCALLS/MEDCALLS/HIGHCALLS/TOPCALLS.

Split/skill calls are reported in the standard reports according to the final disposition of the call.

The presence of the command in a vector enables the calls that are serviced by the vector to be vector-directed. When such a call is answered by an agent, the call is tracked as ACDCALLS/ANSTIME, and it is reported as ACD Calls, Split/skill ACD Calls, and Avg Speed Ans. If the call is also queued to other splits/skills, OUTFLOWCALLS/OUTFLOWTIME is tracked in the first split/skill to which the call queues, and Flow Out is reported (unless the split/skill turns out to be the answering split/skill). DEQUECALLS/DEQUETIME is tracked in the second and third splits/skills if these splits/skills are not the answering split/skill, and the call is reported as Dequeued Calls and Dequeued Avg Queue Time. However, if the second or third split/skill is the answering split/skill, INFLOWCALLS is tracked in the split/skill, and the call is reported as Flow In.

If the call abandons after the command queues the call to a split/skill, ABNCALLS/ABNTIME is tracked for the vector, the VDN, and the first split/skill to which the call is queued. The call is reported as Aban Call and Avg Aban Time. If the call is also queued to other splits/skills, DEQUECALLS/DEQUETIME is tracked in these splits/skills, and the call is reported as Dequeued Calls and Dequeued Avg Queue Time.

BSR status poll calls are not counted as interflows. BSR interflows are now tracked as network interflowed calls (NETCALLS) by the CMS at the receiving switch. The CMS tracks a call's accumulated time-in-VDN as NETINTIME (that is, the NET_TIME value on the CMS at switch C combines the time a call has spent in VDNs at any previous locations, as communicated by ISDN information forwarding. The NETINTIME can be added to the time spent in the local switch to provide reports that include the total time the call has spent in the call center network (e.g., total ASA).

For more information on the database items and reports, see Avaya *Avaya CMS Database Items and Calculations*, 585-780-702, and *Avaya Call Management System Supervisor Version 11 Reports*, 585-210-708.

BCMS interactions

The total number of calls to the VDN that are queued via the command and then answered by an agent within a specified time period is tracked as ACD Calls in the VDN Report. The average time that calls spend in a vector before being connected via the command as an ACD call to an agent is tracked as AVG SPEED ANS in the same report.

There is no added tracking for calls interflowed by BSR. BCMS tracks these calls as outflow in the VDN Report.

Reply-best

Purpose

The **`reply-best`** command is used only in status poll vectors in multi-site BSR applications, where it returns “best” data for its location to the primary vector on the origin switch.

Syntax

```
reply-best
```

Note:

This multi-site BSR command is available only when the Virtual Routing feature is enabled.

Requirements

The EAS feature must be enabled to use the **`reply-best`** command.

Operation

The purpose of the **reply-best** step is to return data for the best resource found by the consider series in a status poll vector to the primary vector in a multi-site BSR application. The status poll vector executes in response to an ISDN call from a **consider** step in the primary vector. Each time the status poll vector executes, the **reply-best** step:

- drops the incoming ISDN call without returning answer supervision
- returns status data to the primary vector via the ISDN DISCONNECT message
- initializes (clears) the "best" data
- terminates processing in the status poll vector

If the incoming call is not an ISDN call, the **reply-best** command will drop the call and log a vector event. No status data will be returned to the origin switch.

If the consider series yields no "best" data, the **reply-best** command will drop the incoming ISDN call without returning answer supervision, terminate vector processing, and return an infinite value for EWT in the DISCONNECT message. A consider series might not produce "best" data for any of the following reasons:

- all resources considered are unstaffed
- no resource considered has an open queue slot
- "best" data has been initialized before execution of the **reply-best** step (because there are no consider steps in the status poll vector or because the vector contains a prior step that initializes "best" data.

For a list of events and vector commands that initialize "best" data produced by consider series, see Events that clear "best" data on page 425.

Answer supervision considerations

The **reply-best** step does not return answer supervision.

CMS/BCMS interactions

Operation of the **reply-best** command is not reported or tracked by the CMS or by the BCMS.

Route-to command

Purpose

Routes calls either to a destination that is specified by digits collected from the caller or an adjunct (**`route-to digits`**), or routes calls to the destination specified by the administered digit string (**`route-to number`**).

Syntax and valid entries

Command	Parameters / Conditions			
route-to				
	digits	**with coverage** _____ **y**es or **n**o		
	meet-me[1]			
	number ____ 0-9, * , # , ~p, ~m, ~s, ~w, ~W, or a leading ~r[2]	**with cov** ___ **y**es or **n**o	**if** ____ digit	
			interflow-qpos	___ <, =, <=	____ 1-9
			unconditionally	
	name1[3]	**with coverage** _____ **y**es or **n**o		
	name2[3]	**with coverage** _____ **y**es or **n**o		
	name3[3]	**with coverage** _____ **y**es or **n**o		

1. This item available with **`meet-me conference`** vectors, only.
2. When the specified number is preceded by **`~r`**, Network Call Redirection is attempted.
3. This item available with Dial by Name feature, only.

Requirements

The **`route-to name`** command requires the Dial by Name feature to be enabled.

Operation

The **`route-to`** command attempts to route a call to a set of digits collected from the caller, from an adjunct, or from the network, or to route to the destination specified by the administered digit string.

For the **`route-to number ... if digit`** command, the call is conditionally routed to a specified destination according to a single digit entered by the caller. If the digit collected in the last **`collect digits`** command matches the specified comparison in relation to the administered digit, the command attempts to route the call to the specified destination.

The destination for a route-to command can be any of the following:

- Internal extension (for example, split/hunt group, station, etc.)
- VDN extension
- Attendant or Attendant Queue
- Remote extension (UDP/DCS)
- External number, such as a TAC or AAR/ARS FAC followed by a public or private network number (for example, 7-digit ETN, 10-digit DDD, etc.)
- Remote Access Extension.
- Service Observing FAC
- Another Avaya switch (when the **`route-to number`** command is followed by ~r and a ten digit number, then Network Call Redirection is attempted)
- Remote Logout of Agent FAC

Note:

The VDN's Class of Restriction (COR) is used for calling permissions.

The **`route-to digits`** command fails if no digits are collected, and vector processing continues at the next vector step.

The **`route-to number ... if digit`** command fails if more than 1 digit is collected or if the digit comparison fails. Vector processing continues at the next command.

The **`route-to number ... if interflow-qpos`** command fails if the call is not in the eligible queue established by the **`interflow-qpos`** condition. Vector processing continues at the next command.

If the **`route-to`** command is successful, vector processing terminates. Otherwise, vector processing continues at the next vector command.

A **`route-to`** step in a vector is treated as cov=n for a covered call regardless of the cov setting on the **`route-to`** command.

If the number expressed in the command is a system extension or an attendant group (and not a VDN), the system considers the step successful if one of the following conditions occurs:

- The endpoint is alerted.
- The endpoint has Call Forwarding or night service (hunt group) enabled, and the (night service) destination forwarded to is alerted.
- The endpoint has off-premises Call Forwarding (UDP hunt night service) enabled, and a trunk is seized.

The system then provides ringback to the caller, and vector processing terminates. However, if the call cannot complete successfully (for example, no idle appearance is available), vector processing continues at the next vector command.

If the number is a VDN extension, the following events occur:

- Vector processing terminates within the current vector.
- If the current VDN is administered with override, the new VDN overrides current VDN information.
- Processing of the vector associated with the VDN extension begins.

If the number is an AAR/ARS FAC plus digits, or if it is a remote UDP extension, standard AAR/ARS processing is performed to select the trunk group and outpulse the digits. If a trunk is seized, vector processing terminates, and the calling party hears feedback provided by the far end. Otherwise, the call cannot complete successfully (because no trunks are available, the FRL/COR is restricted, etc.), and vector processing continues at the next vector command.

If the number is a TAC plus digits, and a trunk is seized, vector processing terminates, and the calling party hears feedback provided by the far end. Otherwise, the call cannot complete successfully (because no trunks are available, the COR is restricted, etc.), and vector processing continues at the next vector command.

If the number is any other number (such as an FAC other than an AAR/ARS or Service Observing), the command is unsuccessful, and vector processing continues at the next vector command.

Abbreviated Dialing special characters can also be used in the number field. Each of these characters instructs the system to take a different action when dialing reaches the point where the character is stored. The characters are as follows:

- **~p** (pause)
- **~w** (wait)
- **~m** (mark)
- **~s** (suppress)
- **~W** (indefinite wait)

Each special character counts as two digits towards the maximum. The maximum number of digits for the command is 16.

The **route-to digits** command can be used to implement an automated attendant function.

Coverage

The optional coverage parameter determines whether coverage should apply during routing. If coverage applies, and if the digits entered are valid, the following occurs:

- Ringback is provided.
- Vector processing terminates.
- Normal termination and coverage are implemented.

Note:

For detailed information about the operation of the route-to command with or without coverage for the different destinations see the table shown in Switch route-to command operation on page 579.

Answer Supervision Considerations

Generally, answer supervision is provided when the destination answers the call. The exception to this involves incoming trunk calls routed to another non-ISDN-PRI trunk. Such calls provide answer supervision when the outgoing trunk is seized.

Feature interactions

When COR checking is applied to a route-to number or route-to digits step, it is the COR of the latest VDN that is used.

The **route-to** command may specify the AAR or ARS access codes. The COR associated with the latest VDN is used to determine the Partitioned Group Number (PGN) time-of-day routing chart. The PGN determines the choice or route tables used on a particular call.

The command may call the AUDIX extension. If this happens, the call is treated as a direct call to AUDIX, and the calling party may retrieve his or her messages.

If the call covers to a VDN, the command supports a remote AUDIX interface to a local hunt group extension that is assigned as a remote AUDIX hunt group. The remote AUDIX hunt group (which has no members and cannot be vector-controlled) forwards the call to the remote AUDIX destination in the same manner as when the hunt group is assigned as a point in the coverage path. A DCS link down condition for a call that covers to a VDN is treated as a direct call to the AUDIX.

If the command is directed to a station with bridged appearances, the bridged appearance button lamps are updated.

The following destinations always result in a failure, and vector processing continues at the next step:

- Controlled trunk group
- Code calling FAC
- Facility test call
- TAAS access code
- Priority access code
- Loudspeaker paging access code
- Station Message Detail Recording (SMDR) account code
- Voice message retrieval access code.

If the command is executed and Direct Outward Dialing (DOD) is in effect, the COR of the latest VDN is compared with the COR of the called facility to determine if the call is permitted. If access is not permitted, the command fails and vector processing continues. In the case where a COR requiring the entry of account codes is assigned to a VDN, and the command is executed by the associated vector, the command is unsuccessful, and vector processing continues at the next step.

The individual extension number assigned to an attendant console can be used as the command's argument.

A call processed by the command can wait in the individual attendant queue and is subsequently removed from vector processing.

The command can access both public and private networks.

If the command dials the attendant, and if the system is in night service, the call routes to the DID Listed Directory Number (LDN) night destination.

The command can place AAR/ARS calls that implement subnet trunking, which is the routing of calls over trunk groups that terminate in switches with different dial plans.

Authorization codes are disabled with respect to routing via VDNs. In other words, if authorization codes are enabled, and a **`route-to`** command in a prompting vector accesses AAR or ARS, and the VDN's FRL does not have the permission to utilize the chosen routing preference, no authorization code is prompted for, and the **`route-to`** command fails.

If the command routes the call without coverage to a display station, the station displays the following: "a = Originator Name to VDN Name."

If the command calls a station that is a member of a pickup group, the call can be picked up by another pickup group member.

Anytime a **`route-to with cov n`** command initiates a call over ISDN-PRI facilities and LAI is optioned, the call will be treated on a Look-Ahead basis. However, if the command is used with the **`coverage yes`** option in effect, unconditional interflow results.

For LAI, the **route-to** command can be considered either a call acceptance vector command or a neutral vector command. The command is considered a call acceptance vector command whenever one of the following is true:

- Command terminates to a valid local destination.
- Command successfully seizes a non-PRI trunk.
- Command execution results in a LAI call attempt, and the call is accepted by the far end switch.

The command is considered a neutral vector command whenever one of the following is true:

- Termination is unsuccessful.
- Trunk is not seized.
- LAI call attempt is denied by the far end switch.

For a call that covers or forwards to a VDN, the **route-to with coverage y** command functions the same way as the **route-to with coverage n** command. For a covered or forwarded call, the coverage option for the command is disabled since such a call should not be further redirected.

A **route-to with cov y** to a station that has call forwarding activated is forwarded.

Service Observing can be initiated with Call Vectoring using the route-to command. See Service Observing routing on page 120 for detailed instructions.

Note:

Appendix I: Operation details for the route-to command on page 579 gives a detailed description of the feature interactions for the **route-to** number with and without coverage command.

CMS interactions

Tracking of the **route-to digits** command varies according to the destination successfully routed to, as follows:

Routed to station or to attendant

Database item	Report heading	Notes
OUTFLOWCALLS/ OUTFLOWTIME	Flow Out Vector Flow Out	1st split
DEQUECALLS/ DEQUETIME	Dequeued Calls Dequeued Avg Queue Time	2nd/3rd splits
INTIME	Avg Time In Vector	
CONNECTCALLS/ CONNECTTIME	Other Calls Connect	answered calls on G3

Routed to trunk

Database item	Report heading	Notes
OUTFLOWCALLS/ OUTFLOWTIME	Flow Out Vector Flow Out VDN Flow Out	1st split
DEQUECALLS/ DEQUETIME	Dequeued Calls Dequeued Avg Queue Time	2nd/3rd splits

Routed to VDN

Database item	Report heading	Notes
OUTFLOWCALLS/ OUTFLOWTIME	Flow Out Vector Flow Out VDN Flow Out	1st split
DEQUECALLS/ DEQUETIME	Dequeued Calls Dequeued Avg Queue Time	2nd/3rd splits
INTIME	Avg Time In Vector	
INFLOWCALLS	Vector Flow In VDN Flow In	new vector new VDN
INTERFLOWCALLS/ INTERFLOWTIME	VDN Flow-Interflow	
INTIME	Avg Time In Vector	

Routed to Split or Hunt Group

Database item	Report heading	Notes
OUTFLOWCALLS/ OUTFLOWTIME	Flow Out	1st split
DEQUECALLS/ DEQUETIME	Dequeued Calls Dequeued Avg Queue Time	2nd/3rd splits
INTIME	Avg Time In Vector	
CALLSOFFERRED		new split
MEDCALLS/ HIGHCALLS		no priority/priority

Note:

For calls that “route to” a split or a hunt group and later intraflow to a station or to an attendant, OTHERCALLS/OTHERTIME are tracked in the vector and in the VDN tables.

Split calls are also shown in the standard reports according to the final disposition of the call.

Calls that route over an ISDN trunk are LAI calls. When a call attempts to “route to” an ISDN trunk (Look-Ahead Interflow), the LOOKATTEMPTS database item is tracked and reported as Look-Ahead Interflow Attempts. If the call successfully routes, LOOKFLOWCALLS/LOOKFLOWTIME are tracked and reported as Look-Ahead Interflow Completions. Interflow always occurs whenever the **`with coverage yes`** option is in effect.

The presence of the command in a vector enables the calls that are serviced by the vector to be vector-directed. When such a call is answered by an agent, the call is tracked as ACDCALLS/ANSTIME, and it is reported as ACD Calls, Split/skill ACD Calls, and Avg Speed Ans. If the call is also queued to other splits, OUTFLOWCALLS/OUTFLOWTIME is tracked in the first split/skill to which the call queues, and Flow Out is reported (unless the split/skill turns out to be the answering split). DEQUECALLS/DEQUETIME is tracked in the second and third splits if these splits are not the answering split, and the call is reported as Dequeued Calls and Dequeued Avg Queue Time. However, if the second or third split/skill is the answering split/skill, INFLOWCALLS is tracked in the split, and the call is reported as Flow In.

If the command directs a call to a destination, the BACKUPCALLS data base item is incremented, and the call is reported as Calls Ans in Backup and Calls Handled/Backup. The Calls Ans in Main report item is calculated by using the algorithm ACDCALLS - BACKUPCALLS.

A call abandoned after the command routes the call to a station or an attendant is tracked in the VDN tables as ABNCALLS/ABNTIME.

BSR interflows are now tracked as network interflowed calls (NETCALLS) by the CMS at the receiving switch. The CMS tracks a call’s accumulated time-in-VDN as NETINTIME (that is, the NET_TIME value on the CMS at switch C combines the time a call has spent in VDNs at any previous locations, as communicated by ISDN information forwarding. The NETINTIME can be added to the time spent in the local switch to provide reports that include the total time the call has spent in the call center network (e.g., total ASA).

For more information on the CMS database items and reports, see *Avaya CMS Database Items and Calculations*, 585-780-702, and *Avaya Call Management System Supervisor Version 11 Reports*, 585-210-708.

BCMS interactions

A call advanced to another position via the command is tracked as outflow in the VDN Report. A call answered by an attendant via the command is also tracked as outflow.

There is no added tracking for calls interflowed by BSR. BCMS tracks these calls as outflow in the VDN Report.

Stop command

Purpose

The **`stop`** command halts the processing of any subsequent vector steps.

Syntax

```
stop
```

Requirements

No special requirements.

Operation

After the **`stop`** command is processed, any calls already queued remain queued, and any wait treatment (for example, silence, ringback, music) is continued. On the other hand, any calls not queued are dropped under the same scenario.

If a TTR is allocated to the call, and if the **`stop`** command is encountered, the TTR is disconnected. However, current call processing continues (that is, the call is not dropped). The caller continues to hear the feedback that was provided before the **`stop`** command was encountered.

Note:

An implicit stop is processed following the last administered command in a vector.

Answer supervision considerations

The command has no effect on answer supervision.

Feature interactions

For LAI, the command is considered a neutral vector command in all cases except when a call is dropped, then it is considered a denial.

CMS interactions

When the command or the end of the vector is encountered, vector INTIME is recorded. This is reported as Avg Time in Vector.

VDISCCALLS database item in the VDN tables pegs call that pass all the way through a vector without ever having been queued.

BCMS interactions

None.

Wait-time command

Purpose

The **`wait-time`** command delays the processing of the next vector step if a specified delay time is included in the command's syntax. Also provides feedback (in the form of silence, ringback, or music) to the caller while the call advances in queue. The Multiple Audio/Music Sources for Vector Delay and Multiple Music Sources on Hold features allow a specified audio or music source to be selected when a call encounters a **`wait-time`** command. See Basic Call Vectoring for more information.

Syntax and valid entries

Command	Parameters / Conditions				
`wait-time`	____	____	**`hearing`**	____________	

	`0-999`
	`0-480`[1]
	`0-8`[1]
		`secs`
		`mins`
		`hrs`
				`music, ringback, silence, i-silent`	

				`or`	
				`audio source ext.`[2]	**`then`** ____________
					`music, ringback, silence, continue`[3]

1. This option is not available for vector administration done through Avaya CMS or Visual Vectors.
2. A valid announcement/music source extension that is defined on the annoucement audio sources form. '
3. The **`continue`** treatment is only valid with Multiple Audio/Music Sources. It indicates that the caller will continue to hear the alternate audio/music source (using and announcement) until another vector command takes effect.

Requirements

Basic Call Vectoring or Call Prompting software must be installed. Also, a music-on-hold port must be provided for the music treatment. Multiple Audio/Music Sources for Vector Delay requires that the Vectoring (G3V4 Enhanced) customer option be enabled.

Operation

The specified feedback is given to the caller, and vector processing waits the specified time before going on to the next step. If the time specified is **`0`**, feedback is provided without any delay in the processing of the next vector step. The feedback given to the caller continues until any one of the following occurs:

- Subsequent vector step (containing **`wait-time`** or **`announcement`**) changes the treatment.
- Vector processing encounters a **`disconnect`** or **`busy`** command.
- Call is routed to another location or to a step that includes an announcement (for example, **`collect digits`**).
- Call is routed to another VDN.
- Call is delivered to a destination (starts ringing at an agent's terminal).
- Switch receives a destination from the ASAI adjunct.
- Vector disconnect timer expires.

Wait times up to 8 hours are allowed for customers who want to use the ASAI Phantom Call feature to track e-mail and fax messages in split queues.

Considerations

When music is indicated as a treatment, it refers to the system music, not an alternate music source.

The tenant number of the active VDN determines the system music the caller hears. You can allow callers to hear a music source other than the one assigned to the active VDN, however, by directly specifying an extension for an audio source with a command such as **`wait-time 30 secs hearing 4301 then music`**.

The **`i-silent`** keyword is for use with adjunct routing-ADR/Lookahead Interflow applications. I-silent provides silence for the specified time, but it is neutral to LAI while all other wait treatments (even with 0 secs settings) provide acceptance.

Multiple audio/music sources

The expanded **`wait-time _ secs hearing <extension> then <treatment2>`** command provides what is known as Multiple Audio/Music Sources wait treatment. The **`<extension>`** option defines an audio or music source that is assigned on the Announcements/Audio Source administration screen.

The source can be interfaced by way of one of the following:

- Analog/DS1/0 (Line Side T1/E1) station ports
- AUX-Trunks
- An Integrated Announcement board

Any of the announcement/audio source types listed above can be configured to do either of the following:

- Play at the beginning with queuing (with the `Queue` field set to y, which is always recommended for call center applications)
- Barge-in operation (Queue field set to "b")

In addition, integrated board announcements can be set to play once ("integrated") or to repeat after each playing continuously (integ-rep). For more information, see “Appendix A: Recorded Announcements”, in *Avaya MultiVantage Call Center Software – Guide to ACD Call Center*, 555-230-716, and the Announcements/Audio Sources screen reference in *Administrator Guide for Avaya MultiVantage Software*, 555-233-506.

The **`<treatment2>`** parameter refers to the treatment that the caller hears after the source specified by **`<extension>`** finishes playing, or the wait-time period expires. The **`<treatment2>`** parameter is also provided if the caller can not be connected to the source. Failure to connect to the source can result from conditions such as:

- source not available - extension/source not assigned
- source disconnected
- source busy
- queuing not assigned

If the **`<extension>`** source is not available when the wait step is reached in the vector one of the following results will occur:

- If **`<treatment2>`** is set to **`continue`**, the caller returns to what they were hearing before the wait-time step.
- If **`<treatment2>`** is set to **`music`**, **`ringback`**, or **`silence`**, vector processing still waits for the specified wait-time while the caller hears **`<treatment2>`**. When the wait-time period expires, the next step in the vector is executed, regardless of the **`<treatment2>`** setting. The caller continues to hear **`<treatment2>`** until a subsequent step changes the treatment. For example, if **`<treatment2>`** is set to **`continue`**, and the **`<extension>`** source (integ-rep or continuous analog/DS1 or AUX-Trunk) is still playing, the caller continues to hear it until a subsequent vector steps changes the treatment.

Note:

If the **`<extension>`** source stops playing or is disconnected, the caller hears silence.

If the audio/music source specified by the **`<extension>`** stops (disconnects) before the wait-time period expires or the caller cannot be connected to that source (source not available), the caller will hear the source specified by the **`then <treatment2>`** segment of the vector. In this case, if **`<treatment2>`** is specified as **`continue`**, then the caller hears silence.

Answer supervision

If the **`music`** or **`audio source`** treatment is included in the command, answer supervision is triggered. If the command is encountered and answer supervision was sent previously, the caller hears the treatment specified in the current command. If, for a CO trunk user, the command with **`silence`**, **`ringback`**, or **`i-silent`** treatment is encountered prior to answer supervision, the caller continues to hear ringback from the CO.

Feature interactions

An implicit wait of 0.2 seconds (with no change in the feedback to the caller) is provided after every seven vector steps if one of these steps does not suspend vector processing. The following steps, if successful, do not suspend vector processing: **`queue-to split`**, **`check split`**, **`goto step`**, **`goto vector`** and **`wait-time 0 seconds`**. The following steps, if unsuccessful, also do not suspend vector processing: **`check split`**, **`route-to`**, **`adjunct routing`**, and **`messaging split`**. The only commands that suspend vector processing are the following: **`announcement`**, **`wait-time > 0, collect digits`**, and **`converse-on split`**.

- Music-on-Hold

  When the command is implemented with music as the treatment, the system-wide music-on-hold feature must be administered. Otherwise, the caller hears silence. When Tenant Partitioning is in use, the tenant number of the active VDN determines the system music that is heard.

  Feedback continues while a subsequent vector step queues for an announcement or for a TTR.

- Look-Ahead Interflow (LAI)

  For LAI, the wait-time command is considered a call acceptance vector command in all cases, except i-silent, which is considered a neutral vector command.

CMS/BCMS interactions

The command is not tracked on the CMS or on the BCMS. Vectors with **`wait-time`** steps are only accessible to CMS if the time unit is administered in **`secs`**.

Appendix B: Vector management and monitoring

To manage your vectors, there are several considerations and tasks of which you need to be aware. This appendix describes these considerations/tasks, including:

Implementation requirements for the Call Vectoring features

The following tables indicate the forms and the hardware required for the following Call Vectoring features:

Basic Call Vectoring Requirements

Form(s)	Hardware
• Vector Directory Number • Hunt Group • Call Vector • Feature Related System Parameters	Announcement capabilities require either: • TN750 Integrated Announcement circuit pack(s), **or** • External announcement facility (analog announcements). Also, each analog announcement requires a port on an analog line circuit pack or on an auxiliary trunk circuit pack. See the *Avaya MultiVantage/Definity Hardware Solutions Guide*, 555-233-200 for a list of available analog circuit packs.

Note:

The TN750 Integrated Announcement circuit pack provides 16 ports for listening to announcements. The system provides for the installation of multiple TN750C Integrated Announcement circuit packs. See “Managing Announcements” in *Administrator Guide for Avaya MultiVantage Software*, 555-233-506 for more details.

Call Prompting Requirements

Form(s)	Hardware
• Vector Directory Number • Hunt Group • Call Vector	Announcement capabilities require either: • TN750 Integrated Announcement circuit pack(s), **or** • External announcement facility (analog announcements). Also, each analog announcement requires a port on an analog line circuit pack. See *Avaya MultiVantage/Definity Hardware Solutions Guide*, 555-233-200 for a list of available analog circuit packs.

G3V4 Enhanced Vectoring Requirements

Form(s)	Hardware
• Vector Directory Number Form • Hunt Group Form • Call Vector Form	Requires no hardware in addition to that required for Basic Call Vectoring.

Advanced Vector routing requirements

Form(s)	Hardware
• Vector Directory Number Form • Hunt Group Form • Call Vector Form	Requires no hardware in addition to that required for Basic Call Vectoring.

Vectoring (Best Service Routing) requirements

Form(s)	Hardware
Single-site BSR	
• Vector Directory Number Form • Call Vector Form	No special hardware required for single-site BSR.
Multi-site BSR	
• Best Service Routing Application Plan form • Vector Directory Number Form • Call Vector Form • ISDN Trunk forms	Multi-site BSR requires no special hardware other than ISDN BRI/PRI connectivity between switches.

ANI/II-Digits requirements

Forms	Hardware
• Vector Directory Number Form • Hunt Group Form • Call Vector Form • Trunk Group Forms • Vector Routing Tables Forms	Requires no hardware in addition to that required for Basic Call Vectoring.

CINFO requirements

Form(s)	Hardware
• Vector Directory Number Form • Hunt Group Form • Call Vector Form • Trunk Group Forms • Vector Routing Tables Forms	Requires no hardware in addition to that required for Basic Call Vectoring.

Look-Ahead Interflow requirements

Forms	Hardware
• Trunk Group Form (ISDN-PRI) • CPN Prefix Table Form	Existing ISDN-PRI hardware can be used for LAI ISDN-PRI connectivity to the receiving switch. Interconnecting facilities must be ISDN-PRI with no interworking (that is, call connections that use both ISDN-PRI and non-ISDN-PRI facilities to complete) for the full capabilities of the feature to be operational. LAI calls that interwork may interflow successfully, but the ability to do so on an intelligent basis will be lost as will the Look-ahead DNIS information. Look-Ahead Interflow calls can connect ISDN-PRI switch-to-switch using private, public, or SDN facilities.

Adjunct Routing requirements

Forms	Hardware
• Hunt Groups • Class of Restriction (for Direct Agent Calls) • Call Vector • Station • Station (ISDN-BRI-ASAI)	**ISDN-BRI Connection** A TN556 ISDN-BRI circuit pack and a TN778 packet control must be in place. The latter provides packet bus control. Also, an adjunct/host processor must be in place to receive the request and select the route. A TN2198 two-wire BRI port circuit pack can be used in place of the TN556. In this case, an NT1 is also required. **MAPD Connection** MAPD hardware is a sandwich of two boards, the TN801 and a Pentium processor, which allows the switch to be connected to Ethernet and TCP/IP networks. The MAPD requires three contiguous slots on the switch: two slots are occupied by the MAPD unit, and the third is reserved for future use. **Packet Bus** The Packet Bus option (G3r only) must be enabled on the Maintenance-Related System Parameters form before associated ISDN-BRI forms and fields can be administered.

Enabling the Vector Disconnect Timer

Call Vectoring provides a Vector Disconnect Timer, which can be set for any amount of time between 1 and 240 minutes inclusive. The timer is enabled by selecting the timer field in the Feature-Related System-Parameters form. The timer is started when vector processing is started. Once the timer runs out, the call is dropped. The timer is canceled when vector processing terminates.

Enabling the timer allows queued calls that have not been answered within a determined amount of time to be dropped. For more information, see *Avaya MultiVantage Call Center Software – Guide to ACD Call Center*, 555-230-716.

Upgrading to a Call Vectoring environment

If you are already equipped with ACD and want to use Call Vectoring, the ACD environment must be upgraded to a Call Vectoring environment. This involves installing VDNs, vectors and hunt groups for the desired Call Vectoring feature(s).

The set of guidelines that follows is intended to serve as a general procedure for upgrading to a Call Vectoring environment.

1. Verify the vector options on the Customer Option Form.
2. Add the VDNs.
3. Evaluate the number of queue slots assigned to each split. Usually, you want to assign enough queue slots to allow all calls processed by Call Vectoring to be queued. See the considerations for Basic Call Vectoring in Appendix C: Considerations for the vectoring features on page 479 for more details.
4. Change hunt-groups to be vector-controlled.
5. Administer the vectors and at least one test hunt group.
6. Test all of the vectors to be installed.
7. Change the trunk groups, night destinations, etc., to use the VDNs.

Changing and testing a vector

Vectors currently being used to process calls should not be changed because changes would have an immediate and uncertain effect on the treatment that the calls are receiving. Instead, a new vector should always be written.

In testing the vector, you should not consider the entire vector at once. Rather, you should first figuratively divide the vector into portions, then test each of these portions until the entire vector is tested.

After the new vector is thoroughly tested, the vector should be brought into service by changing the VDN to point to the new vector.

The set of following guidelines is intended to serve as a general procedure for changing and testing vectors.

1. Check that a current version of the translation data is available.
2. Create a new VDN that points to the new vector. This VDN, which is temporary, is necessary to test the new vector.
3. Administer the new vector. Vector commands should be added and tested, one command at a time, starting with the first command. Be sure that each line is correct before proceeding to the next one.
4. Test the new vector with the new VDN. This ensures the new vector will function correctly when the vector is installed.
5. Install the new vector by changing the old VDN's vector assignment so that the VDNs now point to the new vector. Calls that are already being processed by the old vector will continue to be handled by that vector until the vector terminates vector processing.
6. Once all the calls are handled, remove the old vector and the VDN that was used for testing.

Identifying Links to a Vector

One or more VDNs always point to a vector. In addition, some vectors are linked to other vectors by **`goto vector`** commands or by **`route-to`** commands that point to a VDN. Before you delete or change a vector, you should identify all the VDNs and vectors that will be affected.

The **`list usage vector nnn`** command finds all the VDNs and vectors that send calls to vector ***nnn***, where ***nnn*** is the assigned vector number.

For example, let's say you want to delete vector 3. To determine what other elements of your system send calls to vector 3, enter **`list usage vector 3`** and press **Enter**.

The List Usage Report screen is displayed.

```
list usage vector 3                                          Page 1

                         LIST USAGE REPORT

Used By
Vector                 Vector Number         1                  Step 3
VDN                    VDN Number            58883
```

VDN 58883 points to vector 3. In addition, step 3 in vector 1 sends calls to vector 3. When you delete vector 3, you'll need to change this vector and VDN so they point to a different vector or delete them too.

Finding All Occurrences of a Digit String

A single extension or an external phone number can be used in several elements in a complex vectoring system. When you modify VDNs or vectors, or when you change the phone numbers used in system elements such as **`route-to`** commands or Best Service Routing Plans, the switch allows you to find a specific digit string.

1. The **`list usage digit-string`** ***`(1–16 digits)`*** command finds the specified digit string in vectors, vector routing tables, and Best Service Routing Plans. The digit string can contain the numerals 0–9 and the characters *, #, ~, p, w, W, m, and s.

For example, to find the system elements that route calls to VDN 53338:

2. Type **`list usage digit-string 53338`** and press **Enter**.

   The system displays the List Usage Report screen.

```
list usage digit-string
53338                                    Page 1

                        LIST USAGE REPORT

Used By
Vector                Vector Number        1                 Step  3
Vector                Vector Number        5                 Step  8
Vector                Vector Number        18                Step  4
Vector                Vector Number        37                Step 10
Best Service Routing Plan Number           1             Location  1
Best Service Routing Plan Number           2             Location  3
Best Service Routing Plan Number           5             Location  1
```

Three Best Service Routing Plans and steps in four different vectors route calls to this VDN. If you delete this VDN or assign a different extension, you'll need to update the extension used by these system elements.

Appendix C: Considerations for the vectoring features

This appendix provides various considerations you should bear in mind when using the Call Vectoring features. These considerations are intended to help you get the highest degree of productivity from Call Vectoring. For Look-Ahead Interflow considerations, see Look-Ahead Interflow (LAI) on page 203.

Note:

If EAS is optioned, “skill” replaces “split.”

This appendix includes the following topics:

Displaying VDN names for vector-initiated Direct Agent calls

The Display VDN for Route-to DAC feature improves the efficiency of call center agents who answer vector-initiated Direct Agent calls (DACs) that originate from multiple Vector Directory Numbers (VDNs).

The type of information displayed at the agent station display with a vector-initiated Direct Agent call can be summarized as follows:

- When the Display VDN for Route-to DAC feature is *not* enabled, only the EAS LoginID name for the agent who receives the call is shown.
- When the Display VDN for Route-to DAC feature *is* enabled for such calls, the “active VDN name” associated with the call is shown.

Providing agents with the ability to see the VDN name associated with an incoming call improves agent efficiency and customer satisfaction. For example, if an agent receives incoming trunk calls for different products from three different VDNs, the VDN name displayed by the Display VDN for Route-to DAC feature allows the agent to answer the call as a sales representative of that product. This feature is especially useful when vector-initiated Direct Agent calls route incoming trunk callers to personalized agent providing services for new customers, special product offers, or premier levels of service.

This section contains the following topics:

Operations

The Display VDN for Route-to DAC feature is designed for call scenarios where a VDN-initiated call is routed to a vector where Direct Agent calls are originated by one of the following methods:

- A **`route-to number`** vector step with **`cov`** parameter set to **`y`**, where the `number` field is administerd with a valid EAS loginID extension.
- A **`route-to digits`** vector step with **`coverage`** parameter set to **`y`**, where a **`collect digits`** vector step preceding this step is used to allow the caller to enter the digits for an EAS LoginID extension.

- An **`adjunct routing`** vector step, where a Direct Agent call is originated by the Route Select digit information returned from a CTI application.

The Display VDN for Route-to DAC feature is activated for an incoming trunk call when the call is routed through a VDN that has the `Display VDN for DAC Calls`? field administered to y. When one of the above-listed vector steps routes such an incoming call as a Direct Agent call to an EAS loginID extension, the active VDN name is shown on the called agent station display instead of the called EAS agent's LoginID name. If this call is routed to another EAS agent in the initially-called EAS agent coverage path, the active VDN name will again be shown on the covered-to agent station display, instead of the initially-called EAS agent LoginID name.

Station display formats

If the Display VDN for Route-to DAC feature is activated for an incoming trunk call routed through a VDN to a vector that initiates a Direct Agent call to an EAS agent, the format of the called agent station display appears as one of the following:

```
<Incoming Trunk Name> to <VDN Name>
```

```
<Incoming caller ANI> to <VDN Name>
```

If the Display VDN for Route-To DAC feature is not activated for an incoming trunk call, the called agent station display appears as one of the following:

```
<Incoming Trunk Name> to <EAS loginID extension>
```

```
<Incoming caller ANI> to <EAS loginID extension>
```

Note:

If the EAS agent to which the call is routed by vector-initiated Direct Agent Calling is not available, and the called EAS agent has a coverage path to other EAS agents, the Display VDN for Route-to DAC feature preserves the active VDN name and sends it to the agent station display for a covered-to EAS agent. If the call covers to a normal station extension in the called EAS agent coverage path, the Display VDN for Route-to DAC feature does not apply to the covered-to station display, and the EAS LoginID of the called EAS agent is displayed instead.

Prerequisites

To use the Display VDN for Route-to DAC feature for incoming trunk calls routed through a Vector Directory Number to an EAS agent via Direct Agent Calling, the following prerequisites must be satisfied:

- This feature is available for MultiVantage release R11.1 and later MultiVantage versions.
- The following administration settings are required:
  - The Expert Agent Selection (EAS) feature must be enabled via the System-parameters customer-options form and the Features-related system parameters form.
  - The VDN used to route an incoming trunk call to a vector that initiates a Direct Agent call must have the “Display VDN for DAC Call?” field set on page 2 of the Vector Directory Number form. Also, the Class of Restriction (COR) administered for this VDN must have the “Direct Agent Calling” field set to “y” on page 1 of the Class of Restriction form.
  - The EAS LoginID to which a vector-initiated Direct Agent call is routed must have an administered COR that has the “Direct Agent Calling” field set to “y” on page 1 of the Class of Restriction form.

  For detailed feature administration instructions, see Administering the Display VDN for Route-To DAC feature on page 482.

Administering the Display VDN for Route-To DAC feature

To activate the Display VDN for Route-to DAC feature, the VDN used to route an incoming trunk call must be administered with the “Display VDN for DAC Calls?” field set to “y” The “active VDN name” station display treatment provided the the Display VDN for Route-to DAC feature applies to the initial EAS agent who receives the vector-initiated Direct Agent call, as well as any EAS agents who may be in the coverage path of the EAS agent the call is initially routed to.

To enable the Display VDN for Route-to DAC feature:

1. Log in to the MultiVantage switch administration system.
2. Enter:

   `display system-parameters customer-options.`

3. Go to page 5 of the form, as shown below.

```
display system-parameters customer-options                          Page  5 of  10
                          CALL CENTER OPTIONAL FEATURES

                           Call Center Release: 11.1

                                ACD? y     PASTE (Display PBX Data on Phone)? y
                       BCMS (Basic)? y                          Reason Codes? y
              BCMS/VuStats LoginIDs? y
         BCMS/VuStats Service Level? y             Service Observing (Basic)? y
                  Business Advocate? y     Service Observing (Remote/By FAC)? y
                    Call Work Codes? y              Service Observing (VDNs)? y
      DTMF Feedback Signals for VRU? y                             Timed ACV? y
                   Dynamic Advocate? y                    Vectoring (Basic)? y
       Expert Agent Selection (EAS)? y                Vectoring (Prompting)? y
                            EAS-PHD? y            Vectoring (G3V4 Enhanced)? y
                   Forced ACD Calls? y     Vectoring (ANI/II-Digits Routing)? y
                                           Vectoring (G3V4 Advanced Routing)? y
          Lookahead Interflow (LAI)? y                    Vectoring (CINFO)? y
  Multiple Call Handling (On Request)? y    Vectoring (Best Service Routing)? y
     Multiple Call Handling (Forced)? y                Vectoring (Holidays)? y

         (NOTE: You must logoff & login to effect the permission changes.)
```

4. If the `Expert Agent Selection?` field is set to `n,it must be changed to y.`

Important:

This form can only be changed by installing a new license file to the MultiVantage switch. Contact your Avaya or indirect channel account executive for assistance.

5. Enter:

`change system-parameters features`

6. Go to page 10 of the form, as shown below.

```
change system-parameters features                             Page  10 of  12
                        FEATURE-RELATED SYSTEM PARAMETERS

CALL CENTER SYSTEM PARAMETERS
  EAS
         Expert Agent Selection (EAS) Enabled? y
        Minimum Agent-LoginID Password Length:
          Direct Agent Announcement Extension: 47786              Delay: 0
    Message Waiting Lamp Indicates Status For: loginID

  VECTORING
                    Converse First Data Delay: 0      Second Data Delay: 2
               Converse Signaling Tone (msec): 100         Pause (msec): 70
                     Prompting Timeout (secs): 10
                 Interflow-qpos EWT Threshold: 0
    Reverse Star/Pound Digit for Collect Step? n
          Available Agent Adjustments for BSR? n
  SERVICE OBSERVING
                 Service Observing: Warning Tone? y     or Conference Tone? n
  ASAI
 Call Classification After Answer Supervision? y      Send UCID to ASAI? y
```

7. If the `Expert Agent Selection (EAS) Enabled?` field is set to `n, set the field to y`.
8. Enter:

   change vdn *XXXXX*

   where ***XXXXX*** is the VDN number for which the Display VDN for Route-to DAC feature is to be enabled.
9. Go to page 2 of the form as shown below.

```
change vdn 2004                                                     Page 2 of 2
                                VECTOR DIRECTORY NUMBER
                                     AUDIX Name:
                          Messaging Server Name:
                             Return Destination:

                                BSR Application: 1
                    BSR Available Agent Strategy: 1st-found
                      Delay ISDN CONNECT message? n
                         Observe on Agent Answer? n
     Delay ISDN CONNECT Message on Agent Answer? n
                           Forward Held-Call CPN? n
                    Display VDN for Route-To DAC? y
```

10. Set the `Display VDN for Route-To DAC?` field to **y**.

Creating vectors that use the Display VDN for Route-to DAC feature

You can administer a vector in several different ways to utilize the Display VDN for Route-to DAC feature.

Note:

For any of the vector examples shown below, if an incoming trunk call is routed through a `VDN with the Display VDN for Route-to DAC?` field set to y, the Direct Agent call is activated with the VDN Display for Route-to DAC feature.

Using collect digits and route-to digits commands

The following vector example shows how to:

- Use a **`collect digits`** vector step to prompt a caller to enter digits for a valid EAS agent loginID extension
- Use a **`route-to digits`** vector step to route the call to an agent as a Direct Agent call:

:

```
wait-time 0 secs hearing ringback
collect 5 digits after announcement 3001
go to step 5 if digits < > 1????
route-to digits with coverage y
announcement 3002
goto step 2
```

Using route-to number commands

The following simple vector uses the **`route-to number`** vector step to originate a Direct Agent call to an EAS LoginID extension:

```
wait-time 0 secs hearing ringback
route-to number 85103 with cov y
```

Using adjunct routing commands

You can also originate a Direct Agent call with a vector that includes an **`adjunct route`** vector step. When an incoming trunk call is routed through a VDN to a vector that includes an **`adjunct route`** vector step, vector processing treats this step like a **`route-to number with cov`** set to **`y`** vector step.

The following vector uses the adjunct route vector step to originate a Direct Agent call. In this example, the CTI application would be designed to route the call as a Direct Agent call in a Route Select ASAI message.

```
1. wait 0 secs hearing ringback
2. adjunct route link 3
3. wait 30 secs hearing ringback
4. announcement 3501
5. disconnect
```

Interactions with other MultiVantage features

Interactions of the Display VDN for Route-to DAC feature with other MultiVantage features include the following:

Call Coverage – When the “Display VDN for Route-to DAC” feature is activated for a call, and a vector-initiated Direct Agent call is made to an EAS agent having a coverage path that has other agents as coverage points, the active VDN name associated with the call is displayed on a covered-to agent’s station display instead of the originally-called EAS agent’s LoginID extension.

Call Forwarding – Display VDN for Route-to DAC has no impact on the Call Forwarding feature.

Station Conference/Transfer – When an EAS agent transfers or conferences a vector-initiated Direct Agent call that has the Display VDN for Route-to DAC feature activated to another agent or station user, the station display of the answering agent or station does not show the active VDN name that was previously displayed for the call. This is consistent with the existing station display treatment for transferred or conferenced calls that have a VDN name shown as the *to* party for a call.

VDN Override – Active VDN name station display rules for the VDN Override feature are applied to the Display VDN for Route-to DAC feature. For example, if an incoming trunk call is routed through a VDN where the VDN Override feature is enabled, and the call is routed to a second VDN by a “route-to number” vector step where the **Display VDN for Route-To DAC?** option is set to **y**, the station display for an EAS agent that receives a subsequent vector-initiated Direct Agent call shows the second VDN’s name for the call instead of the called EAS agent’s LoginID extension.

Redirect on No Answer (RONA) – The Display VDN for Route-to DAC feature is activated only for vector-initiated Direct Agent call to an EAS LoginID extension. When the RONA timer expires after the call is not answered, one of the following results occurs:

- If subsequent vector processing again routes the call to an EAS LoginID extension by means of the Direct Agent Calling feature, and the Display VDN for Route-to DAC feature is enabled, the active VDN name is shown on the covered-to agent station display.
- If subsequent vector processing again routes the call to an EAS LoginID extension by means of the Direct Agent Calling feature, and the Display VDN for Route-to DAC feature is not enabled, then the EAS LoginID for the covered-to agent is shown on their station display.

Audix Messaging for EAS Agents – The Display VDN for Route-To DAC feature has no interaction with Audix Messaging for a vector-initiated Direct Agent call that is routed to an EAS agent and subsequently covers to the agent's Audix mailbox.

Adjunct Routing – If a call is routed through a VDN having the **Display VDN for Route-to DAC?** feature set to **y**, and an **`adjunct route`** vector step is executed that results in a Direct Agent call to an EAS agent, the “active VDN name” is displayed on the routed-to agent's station display instead of the called EAS agent's LoginID.

Transferring calls to VDNs

Care needs to be taken when writing a vector to which callers will be transferred. This is especially true if the vector manipulates or tests data that is delivered with the incoming call, such as ANI, II-digits, or CINFO digits.

To understand why care is needed, it is necessary to understand how a transferred call is treated. There are three main steps in a call transfer.

1. The transferring party hits the transfer button. The caller is put on hold. A second call is created with the transferring party as the originator.
2. The transferring party dials the VDN extension. Vector processing starts. The transferring party, not the caller, hears the initial vector provided feedback, if any.
3. The transferring party hits the transfer button for the second time. The two calls merge. The transferring party is dropped from the call. The caller becomes the originator of the new call. The caller now begins to receive vector provided feedback.

Between transfer steps 2 and 3 there is always a small but finite amount of time during which it is the transferring party who is connected to the vector. Any testing of ANI, II-digits, or CINFO digits during this time window applies to the transferring party and not to the caller. For this reason, it is recommended that vectors not start with an ANI, II-digit, or collect cdpd/ced step. Insert a delay of sufficient length to allow the transferring party to complete the transfer.

A delay is not required before a **`collect x digits after announcement`** step because a collect announcement is restarted for the caller when the transfer is complete.

VDN Return Destination

The VDN Return Destination feature allows an incoming trunk call to be placed back in vector processing after all parties, except the originator, drop. This feature is activated through switch administration of the VDN form. This feature is included in the Avaya Call Center Deluxe package and the Avaya Call Center Elite package.

A field on the VDN form allows the user to enter a VDN extension as a Return Destination. In this section, the VDN which has the Return Destination field administered will be called the “VDN with this feature active.” The Return Destination VDN (the one specified in the new field) will be referred to as the “Return Destination.”

Every incoming trunk call that is processed through a VDN with this feature active will be placed back in vector processing when all parties on the call, except the originator, drop. For this feature, the “originator” is the incoming party that originated the call at the time the call entered the VDN with this feature active.

Note:

Incoming calls on DCS ties do not go to VDN Return Destination.

The VDN that the call will be placed in (when the originator is the only remaining party) is determined by the “Return Destination”. This VDN may be the same or different than the original VDN.

This feature is used to keep the call active and give the caller the opportunity to signal the need for sequence dialing (by entering a “#”). There are two ways this can happen:

1. When the destination drops on its own (after having answered), the call will go to the Return Destination which will have a **`collect digits`** vector step. This step will try to collect the “#” sign entered by the caller.
2. When the call is not answered, the caller enters the “#” to request sequence calling (this “#” will be collected by the ASAI-Requested Digit Collection feature). This “#” is reported to the adjunct. The adjunct requests the `third_party_drop` (or `third_party_end_call`) for the destination, and at that point the call goes to the Return Destination.

The VDN Return Destination and ASAI-Requested Digit Collection features may be used independently, with the following rules:

1. If there is no ASAI request to collect digits, but a Return Destination is provided: when all parties, except the originator, drop, the switch will route the call with only one party active (the caller) to the Return Destination. At this point, the call enters vector processing for the VDN specified by the Return Destination.

   The caller will keep returning to this same return destination indefinitely until either the caller hangs up or a busy or disconnect vector step is executed. Once a call leaves vector processing for the first time, the return destination will never be changed.

2. If a request is made to collect digits but there is no Return Destination provided: the switch will collect the digits and pass them on to the ASAI adjunct. It will be up to the adjunct to take action. However, if the action taken by the adjunct is to drop one party on the call, the switch will drop the other party as well and clear the call (it cannot retain a call with only one party, if there is no Return Destination for further processing).

User scenario — remote access with host provided security

A customer may use the VDN Return Destination feature to provide a more flexible remote access feature together with host-based call security. The remote user/caller does not have to call back into the switch when multiple destinations need to be reached nor does the caller have to enter his/her identification every time a new destination is desired.

This system consists of three VDN/vector pairs. The first VDN uses the vector shown in The following example.

Sample vector for remote access

```
1. collect 6 digits after announcement 1001 ("Please enter
     your identification number and password followed by # sign")
2. adjunct routing link 1221
3. wait-time 6 seconds hearing silence
4. disconnect after announcement 1003 ("We are sorry, but we are
     experiencing technical difficulties at this time, please try
     again later")
```

In this scenario, a remote caller calls into the switch by dialing the first VDN. The vector shown above prompts the caller to enter an identification number and a password that will be passed, via the adjunct routing vector command, to the host for validation. The host can keep track of invalid attempts or decide to de-activate or activate certain identification numbers based on customer set criteria. If the host is not available, the call will be disconnected after an announcement (vector step 4 above).

Sample return destination vector with disconnect

```
1. collect 16 digits after announcement 1002 ("Please enter
     the telephone number of your destination, followed by # sign")
2. adjunct routing link 1222
3. wait-time 6 seconds hearing silence
4. disconnect after announcement 1003 ("We are sorry, but we are
     experiencing technical difficulties at this time, please try
     again later")
```

If the ID and password are valid, the adjunct specifies a route to the second VDN, which uses the vector shown above in Sample return destination vector with disconnect on page 490. The switch collects digits for the destination that the caller wants to reach (vector step 1 above). The host receives the number entered by the caller (vector step 2 above) and validates the entered number to check if the caller is allowed to reach the specified destination. If so, the host routes the call to the destination. After the called destination disconnects from a call, the caller can remain on the line to be connected to the Return Destination, which points to the same vector.

Note:

If the ID or password entered at the first VDN is invalid, then the call can be routed to a third VDN. The vector for this VDN (not shown) consists simply of a **`disconnect after announcement`** step with an appropriate announcement. The invalid call attempt is logged.

The caller, once connected to the Return Destination, can enter a second destination/phone number to connect to. The host performs the same validation on the destination number as in the first destination and routes the call as appropriate (destination entered by caller or alternate destination). Note that the host can also provide reports on all the destinations and times reached by each remote user.

In the Return Destination vector, it is recommended that the first vector command give the caller the opportunity to disconnect from the call rather than immediately routing the call to some destination. If the call was immediately routed and then the caller decided to hang-up, the destination that the call was routed to would ring, alerting the called party, but then no one would be on the line at the other end (this could be confusing to customers, and could be misinterpreted as a problem with the feature). Vector commands such as **`wait-time`**, **`collect after announcement`**, and **`announcement`** can provide the caller with the opportunity to disconnect before the call is routed. As an example, an **`announcement`** command with the recording “Please hang-up to end your call, or remain on the line if you wish to place another call” instructs the caller to disconnect, before the call is routed.

User scenario — saving in trunk facilities between call centers

You can also use VDN Return Destination to return a call to a local agent after the call is transferred to a remote destination (call). This eliminates the need for the remote agent to transfer the caller back to a local agent and will save in switch trunk facilities, since each time the call is transferred back to a local agent an additional trunk is being used by the call.

For example, calls can be received at the local call through a VDN that has the return destination administered. These calls are delivered to an agent on the local switch. If the local agent transfers the call to a remote destination (because the caller needed to talk to an agent on the remote switch), the call returns to the Return Destination after the remote switch drops the call. The remote switch agent must inform the caller to remain on the line after they are finished and the remote agent just needs to disconnect from the call (hang up).

The Return Destination for this scenario should include an **announcement** vector command at the beginning to inform the caller to disconnect from the call, if they do not want to be reconnected to an agent on the local switch. A sample Return Destination vector is shown in the following example.

Sample return destination vector with announcement

```
1. announcement 1004 ("Please remain on the line, if you want
     to talk a to another representative")
2. queue-to split 101 pri m
3. announcement 1005 ("All our representatives are busy,
     please wait")
4. wait-time 60 secs hearing silence
5. goto step 3 if unconditionally
```

Appendix D: Troubleshooting vectors

This appendix serves as a troubleshooting guide for Call Vectoring. The sections include the following topics:

Criteria for success/failure of call vectoring commands

The following table summarizes the success and failure criteria for various vector commands. Before you write or evaluate vectors, it is important to understand the information in this table.

Note:
If EAS is enabled, “skill” replaces “split.”

Call vectoring command success/failure criteria	
`adjunct routing`	
Fails if any of the following are true: • VDN’s COR does not permit routing to the adjunct-supplied destination. • TAC/ARS/AAR code is invalid. • Specified agent is not logged into the specified split for a direct agent call. • Local extension is not in the dialplan. • Invalid number was dialed.	Stop **`wait-time`** or **`announcement`** step (if present). Then continue vector processing with the next sequential step.
Otherwise, succeeds.	Route the call and provide feedback.
`announcement`	
Fails if specified announcement is not administered, not recorded, or busied out.	Continue vector processing with the next sequential step.
Otherwise, succeeds.	Play the announcement, then continue at the next sequential step.
`busy`	
Always succeeds. (CO without answer supervision trunk callers will not hear the busy tone.)	Exit vector processing, then play the busy tone for 45 seconds before dropping the call. (Unanswered CO trunk calls receive 45 seconds of ringback.)

Call vectoring command success/failure criteria (continued)	
`check split`	
Fails if any of the following are true: ● Vector conditional is false. ● Split's queue is full. ● Split is not vector-controlled. ● Call is already queued at the specified priority to the specified split. ● Call is already queued to three different splits.	Continue vector processing with the next sequential step.
Otherwise:	
Succeeds, and the call is terminated to an agent.	Exit vector processing, and pass control to call processing.
Succeeds, and the call is queued or requeued in the specified split at the specified priority.	Continue vector processing with the next sequential step.
`collect-digits`	
Fails if any of the following are true:	
Call originates from an outside caller who is not using a touch-tone telephone.	Call Prompting timer takes effect, command times out, and vector processing continues at the next vector step.
No TTR is in the system, or the TTR queue is full.	Continue vector processing at the next step.
Caller enters fewer digits than the maximum specified.	Call Prompting timer takes effect, command is terminated, and any digits collected prior to the timeout are available for subsequent processing.
Otherwise, succeeds.	Continue vector processing at the next step.

Call vectoring command success/failure criteria (continued)	
`consider locations`	
Fails if any of the following are true: • No BSR application administered in active VDN. • Location not administered in BSR application. • Status Poll VDN number not administered in BSR application. • Status Poll VDN number is invalid. • Status Poll fails because all trunks are busy.	Continue vector processing with the next sequential step.
Otherwise:	
Succeeds, but takes no action if polling of specified location is suppressed.	Continue vector processing with the next sequential step.
Succeeds, and place status poll call to the status poll VDN.	Suspend vector processing until status poll response received.
`consider split`	
Fails if any of the following are true: • VDN skill (1st, 2nd, 3rd) is used in consider step but not administered for active VDN. Otherwise: Succeeds, and the status of the local split is evaluated.	Continue vector processing with the next sequential step.
`converse-on split`	
Fails if any of the following are true: • Converse split queue is full. • Converse split is not vector-controlled. • Auto-available split is in effect, and all agents are logged out by Redirection on No Answer (RONA).	Continue vector processing with the next sequential step.
Otherwise: Succeeds, call is delivered to the converse split, and (if administered) digits are outpulsed to the VRU. The caller is connected to the VRU, the voice response script is executed, and (if necessary) digits are outpulsed to the switch.	Continue vector processing with the next sequential step.

Call vectoring command success/failure criteria (continued)	
`disconnect`	
Always succeeds.	Play the announcement (if specified). Then drop the call.
`goto step and goto vector`	
Fails if the step condition is not met.	Continue vector processing with the next sequential step.
Succeeds if the step condition is met.	**`goto step`** - continue vector processing with the destination step **`goto vector -`** continue vector processing with the first nonblank step of the destination vector.
`messaging split`	
Fails if any of the following are true: ● Specified split is not an AUDIX split. ● Specified extension is invalid. ● Messaging split queue is full. ● Messaging split is not vector controlled and has no working agents (none logged in or all in AUX work mode). ● Communications link with the messaging adjunct is inaccessible.	Continue vector processing with the next sequential step.
Otherwise, succeeds.	Terminate vector processing.

Call vectoring command success/failure criteria (continued)	
`queue-to split`	
Fails if any of the following are true: • Split's queue is full. • Split is not vector-controlled. • Call is already queued at the specified priority to the specified split. • Call is already queued to three different splits. Otherwise:	Continue vector processing with the next sequential step.
Succeeds, and the call is terminated to an agent.	Exit vector processing, and pass control to call processing.
Succeeds, and the call is queued or requeued in the specified split at the specified priority.	Continue vector processing with the next sequential step.
`reply-best`	
Fails if any of the following are true: • Incoming call is not ISDN • Incoming trunk group is not administered for shared UUI or for QSIG Supplementary Service b.	Drop the call.
Otherwise: Succeeds and returns status data of best resource found in consider series.	Drop the call.
`stop`	
Always succeeds.	Exit vector processing. Control is passed to normal call processing. Any queuing or treatment in effect remains in effect. Call is dropped if not queued.
`wait-time`	
Always succeeds.	Connect the specified treatment and pass control to the delay timer. Any feedback is continued until other feedback is provided.

Unexpected feature operations

The following table indicates and explains unexpected operations within Call Vectoring that you may encounter.

Unexpected feature operations	
Customer observations	**Causes**
General Vector Processing	
Vector stuck	1000 steps executed (3000 with enhanced LAI). No default treatment in the vector.
Audible feedback lasts longer than the delay interval.	Last vector step. Queuing for an announcement. Queuing for a touch-tone receiver for a **`collect digits`** step.
Look-Ahead Interflow	
Agent receiving phantom call.	Agents on both switches become available simultaneously. Avoid by including at the beginning of the receiving switch vector a short **`wait-time`** or **`announcement`** step. Also, use the interflow-qpos conditional (see How enhanced LAI works on page 213).
Remote agent receiving phantom calls when vectoring uses qpos conditional.	Interflow-qpos threshold may be set too low.
No Look-Ahead Interflow attempts accepted.	No trunks. PRI network failure. Insufficient FRL.

Unexpected feature operations (continued)	
Customer observations	**Causes**
All Look-Ahead Interflow attempts accepted.	Look-Ahead Interflow attempts are interworking off of one of the following: Interworking off of the PRI network Receiving vector not designed for conditional acceptance **`route-to with coverage yes`** command was used to interflow Look-Ahead Interflow not optioned at the receiving switch.
Look-Ahead DNIS name not displayed or no collected digits received	LAI IE or VDN Name (Shared UUI) not forwarding with call. Trunk group settings are not administered to support this data. For more information, see Information Forwarding on page 151.

Unexpected command operations

The following table indicates and explains the unexpected operations the customer may encounter in using the Call Vectoring commands.

Unexpected command operations	
Customer observation	**Cause**
`adjunct routing`	
Step skipped (that is, default treatment).	Invalid link extension. No trunks available. COR/FRL restricted. Timeout. (Application did not respond within the time specified in the **`wait-time`** command and/or within the time length of the recorded announcement.) Digit string inconsistent with networking translation. ASAI link down. Invalid route destination returned from adjunct.
Busy tone.	Busy local destination has no available coverage points.
Network reorder or intercept.	Digit string supplied by adjunct inconsistent with public network translation. Digit string inconsistent with networking translation.
Intercept or reorder tone heard.	Vector processing succeeded routing off switch, but a problem has occurred before routing to its final destination.
All trunks busy on a quiet system.	Two switches treating each other as backup switch.
Step skipped.	Port Network (PN) link down.

Unexpected command operations (continued)	
Customer observation	**Cause**
`announcement`	
Announcement not heard.	Announcement board not present. Announcement not administered. Announcement not recorded. Announcement being rerecorded. All ports busied out. Announcement restore in progress. Link to announcement circuit pack down.
Extra delay before hearing announcement.	Announcement queue full.
	All integrated announcement ports busy.
	Analog announcement busy.
Vector processing stops.	Analog announcement does not answer.
Listening to silence after announcement.	Announcement is the last step.
Incomplete announcement.	Agent becomes available.
	Previous adjunct routing step succeeds.
`busy`	
Ringback heard instead of busy tone.	Unanswered CO trunk.
`check`	
Call does not enter queue or terminate to agent.	Step condition not met.

Unexpected command operations (continued)	
Customer observation	**Cause**
`check` and **`queue-to`**	
Call does not enter queue or terminate to agent.	Queue length specified on the hunt group screen has been exceeded. Invalid split. Split not vector-controlled. Already queued to three different splits. No queue. Queue or check status indicates space when queue is full due to direct agent calls. Best keyword is used but consider series is not defining “best” data.
Call apparently answered in wrong order.	Call being requeued at different priority. Call superseded by higher priority call, including direct agent call.
Call is not routed to remote best location.	No trunk available.
`collect digits`	
Announcement not heard while waiting for digits, but network billing indicates that the call was answered.	Announcement board not present. Announcement not administered. Announcement not recorded. Announcement being rerecorded. All ports busied out. Announcement restore in progress. Dial ahead digit exists.

Unexpected command operations (continued)	
Customer observation	**Cause**
`collect digits` (continued)	
Collect step and announcement skipped.	TTR not in system. Link to PN that has TTR is down. TTR queue full.
Delay before hearing announcement.	All TTR ports busy, but space in queue. Announcement queue full. All integrated announcement ports busy. Analog announcement busy.
Vector stuck.	Analog announcement does not answer.
Dial-ahead digits not recognized.	Dial-ahead digits entered prior to first collection step. Call has been transferred. LAI attempt has been made. TTR has been released. 24 digits have already been provided. Call Prompting timeout since the last digit was entered.
Vector processing halted at collect step; announcement heard again upon return.	Call put on hold, transferred, or conferenced.
Insufficient digits collected; call routed to intercept.	Caller dialed # too soon. Caller dialed * without reentering correct digits. Call Prompting interdigit time-out.
Caller information button denied.	No digits were collected. Display not in Normal mode.

Unexpected command operations (continued)	
Customer observation	**Cause**
Collect announcement not heard and first collected digit incorrect.	System does not contain all TN748C Vintage 5 (or later) circuit packs.
Incomplete announcement.	Agent becomes available. First digit dialed.
`consider`	
Local split/skill best (in Primary vector or Status Poll vector)	If split/skill number is correct, split or skill has no agents logged in, no queue slots available, or all agents are in AUX work.
Remote location is never best	No BSR application plan assigned to Primary VDN. Location number not assigned in application plan. Missing routing number for Status Poll VDN. No vector assigned to Status Poll VDN. Step in Status Poll vector is initializing “best” data before reply-best step.
`converse-on split`[1]	
VRU script not executed.	Queue full. No queue. Invalid split. Split not vector-controlled. VRU down.
“Ani” digits not passed.	ANI not available.
“Qpos” digits not passed.	Call not queued to a nonconverse split.
No data returned from VRU.	No TTRs available.
VRU script terminated prematurely.	Agent becomes available. VRU script attempted to transfer the call.
Wait digits not passed	Call not queued or no working agents in splits where call is queued.
`disconnect`	
Announcement not heard.	Announcement board not present. Announcement not administered. Announcement not recorded. Announcement being rerecorded. All ports busied out. Announcement restore in progress.

Unexpected command operations (continued)	
Customer observation	**Cause**
`disconnect` (continued)	
Extra delay.	Announcement queue full. All integrated announcement ports busy. All analog announcements busy.
Vector stuck.	Analog announcement does not answer.
`goto step`	
Branch is not made to the specified step.	Step condition not met. System time not set.
`goto vector`	
Branch is not made to the specified vector.	Step condition not met.
Vector stuck.	Goto vector with no steps or with all failed steps.
`messaging split`	
Vector stuck (with ringback).	Extension unknown to AUDIX.
Step skipped, no message left.	AUDIX link down. DCS link to remote AUDIX down. All DCS trunks busy. Queue for AUDIX voice ports is full.
Vector stuck (with busy).	Remote AUDIX link down.
Messages not found.	Message extension is none (message is left for VDN that accessed the vector).
Delay before AUDIX answers.	All AUDIX ports busy, but space in queue.
Busy tone. Step skipped.	Queue for AUDIX voice ports is full. Split not AUDIX split anymore.

Unexpected command operations (continued)	
Customer observation	**Cause**
`reply-best`	
Status poll VDN/vector not processing any calls	Incoming call not ISDN. No application plan defined for BSR application. Status Poll VDN routing number missing from or wrong in application plan.
`route-to`[2]	
Step skipped (that is, default treatment).	Invalid local extension. No trunks available. COR/FRL restricted. Digit string inconsistent with networking translation. Busy local destination (route to digits without coverage and route to number). No digits collected. Step condition not met.
Network reorder.	Digit string inconsistent with public network translation.
Intercept or reorder tone heard.	Vector processing succeeded routing off switch, but a problem has occurred before routing to its final destination.
All trunks busy on a quiet system.	Two switches treating each other as a backup switch.
`stop`	
Call dropped.	Call not queued when vector processing stops.

Unexpected command operations (continued)	
Customer observation	**Cause**
`wait-time`	
Audible feedback longer than delay interval.	Queuing for an announcement or for a TTR. **`Stop`** command executed.
Audible feedback shorter than delay interval.	Agent becomes available. Previous **`adjunct routing`** step succeeds.
Music not heard.	No music port administered. Music source disconnected or turned off.
Alternate audio/music source not heard	Announcement board not present. Audio/Music source not administered. Audio/Music source not recorded. Audio/Music source being rerecorded. All ports busied out. Announcement restore in progress.

1. Refer to the Converse command debugging section later in this appendix for more details on converse-on command debugging

2. Complete operation details for the route to commands are presented in Appendix I: Operation details for the route-to command on page 579.

Converse command debugging

The following table is intended to help your troubleshooting efforts with the **`converse-on`** command.

Note:

Refer to Appendix J: Call flow and specifications for converse – VRI calls on page 585 for details on the call flow for converse-VRI calls.

Converse command debugging

Symptom	Cause	Analysis
Placing a call:		
Converse step skipped.	VRU down (RONA).	Vector event.
	Split queue full	Vector event.
Call stuck in converse.	VRU port doesn't answer, RONA not used.	Check split administration.
	VRU down, RONA leaves call in queue.	Check split status.
Data passing:		
First set of digits not collected.	Converse first delay too short.	Check administration.
	No ANI available.	Vector event.
	No digits collected.	Vector event.
	Call not queued (qpos).	Vector event.
	Expected wait time not available	Vector event.
	VRU timed out awaiting first digit.	VRU error log/trace.
	VRU first digit timeout too short.	Check VRU script.
		Check converse first data delay.
	Faulty hardware.	Diagnostics

Converse command debugging (continued)

Symptom	Cause	Analysis
Second set of digits not collected.	VRU digit count on first prompt in VRU script does not include "#."	Check VRU script.
	Converse second delay too short.	Check administration.
	No ANI available.	Vector event.
	No digits collected.	Vector event.
	Call not queued (qpos).	Vector event.
	Expected wait time not available because call is not queued or the splits/skills that the call is queued to are not staffed	Vector Event
	VRU timed out awaiting first digit.	
	VRU error log/trace.	
	VRU first digit timeout too short.	Check VRU script. Check converse second data delay.
	Inter-digit timeout too short on first prompt and collect.	Check VRU script.
	Faulty hardware.	Diagnostics.
Digits incomplete.	Converse data delay too short.	Check administration.
	Faulty hardware.	Diagnostics.
Second set of digits is the same as the first digits passed.	VRUs first prompt timed out.	Check administration.
	Faulty hardware.	Diagnostics.

Converse command debugging (continued)

Symptom	Cause	Analysis
Data return:		
No digits returned to the switch.	Flash not recognized by switch.	VRU error log/trace.
		Check flash timing on VRU.
	Converse data return FAC not administered.	Check administration.
	VRU does not return FAC.	VRU script. Transfer attempt vector event.
	VRU returns incorrect FAC.	VRU script. Transfer attempt vector event.
	Digit timeout during FAC.	Transfer attempt event.
	Converse data return FAC overlaps with other entries in the dial plan	Check dial plan.
	Faulty hardware.	Diagnostics.
Not all digits returned to the switch.	Digit timeout after FAC.	None unless VRU logs being dropped by the switch.
	Overflow of Call Prompting buffer	Vector Event.
	Faulty hardware.	Diagnostics.
Collect announcement not heard.	Too many digits returned by VRU.	Check VRU script.
	Faulty hardware.	Diagnostics.

Tracking unexpected events

You can display unexpected events related to Call Vectoring and Meet-me Conference. When you have corrected each problem, then you can clear events from the error log. An event is an error that results from resource exhaustion, from faulty vector programming, or from incorrect user operation rather than from a switch software error. For example, failures involving the **`route-to`** command are usually due to an invalid extension entered by the user.

By displaying events, you can diagnose and correct each problem, as indicated by its corresponding event number, and eliminate the need for a technician to make on-site visits to do the same.

The following sections explain how you can troubleshoot by tracking unexpected events.

Display events criteria

Use the **`display events`** command to access the EVENT REPORT screen. Use the fields on this screen to specify the event report criteria.

```
display events                                          Page   1 of   1   SPE B
                                  EVENT REPORT

      The following options control which events will be displayed.

         EVENT CATEGORY

            Category: meetme

         REPORT PERIOD

            Interval: a      From:   /  /  :   To:   /  /  :

         SEARCH OPTIONS

                            Vector Number:
                               Event Type:
                                Extension: 36090
```

The following table describes the fields used with the **`display events`** command.

Field	Description
Category	Enter **`denial`**, **`meetme`**, **`vector`**, or **`all`** to specify the type of event you want to display.
Interval	Select the time period for which you want to display events. Enter **h** (hour), **`d`** (day), **`w`** (week), **`m`** (month), or **`a`** (all).
From/To	Enter the date and time of day when you want to start and end the search.
Vector Number	Enter a specific vector number to report on. When the Category field is set to `meetme`, this field is ignored.
Event Type	Enter a specific event type to report on. If this field is blank, events for all types are reported.
Extension	Enter a specific extension or VDN to report on. If this field is blank, events for all extensions are reported.

Display events report

After you have entered your report criteria, submit the command by pressing **Enter**. The following screen shows examples of events.

```
display events

                                 EVENTS REPORT

Event Event                      Event     Event      First        Last     Evnt
Type  Description                Data 1    Data 2     Occur        Occur     Cnt

90    Wait step music failed      3/1     2A2      02/12/15:42 02/13/09:40  255
112   Converse no prompt digits   3/2     2A2      02/12/15:42 02/13/09:40  255
56    Call not in queue           8/1     28B      02/12/15:43 02/13/09:40  255
220   EWT call not queued         8/2     28B      02/12/15:43 02/13/09:40  255
150   Invalid hunt group          8/3     28B      02/12/15:43 02/13/09:40  255
56    Call not in queue           8/5     28B      02/12/15:43 02/13/09:40  255
```

The following table describes the information displayed in the event report.

Column	Description
Event Type	Displays a unique number that identifies the type of event that occurred. These are explained in more detail in Summary of events on page 515.
Event Description	Displays a brief explanation of the event.
Event Data 1	Displays the following data: ● *<number1>/<number2>* (for example, 12/5), where *<number1>* is the vector number associated with the vector event, and where *<number2>* is the step number associated with the vector event. ● *Split<number>* (for example, Split 89), where *<number>* is the split number associated with the vector event. ● For Meet-me Conference events, this is the port ID of the user associated with the event.
Event Data 2	Displays the following data: ● Additional data encoded as a hex number (for example, 4C). This number serves as a call identifier. If two or more events with an identical identifier occur at about the same time, it can be concluded that the events were caused by the same call. ● For Meet-me Conference events, this is the VDN of the Meet-me Conference used during the event.
First Occur/Last Occur	Displays the date and time the event first occurred and the date and time the event last occurred.
Evnt Cnt	Displays, up to 255, the total number of vector events of this type that have occurred.

Summary of events

The following table provides a list of events, the brief description that displays on the screen, and a full explanation of the event.

Summary of events

Event type	Event description	Explanation
1	Call dropped; call not queued at stop step.	Vector processing ended without the call being queued to a split and, as a result, the call cannot be answered. This implies that some default condition was not programmed or that the vector was designed to not always answer the call. Also, call was subsequently dropped.
2	Vector with no steps	The call encountered a vector with no steps administered.
3	1000 step executed	This can occur due to the following: Incorrect vector programming (for example, including a series of **`goto`** steps that point to one another) Excessive repetition of a programmed loop during a single call (for example, recurring announcement-wait loop)
4	Administration change	The administration of this step occurred while the step was being executed. The call flow for this call is unpredictable. Vectors should not be changed while calls are active.
5	Call dropped by vector disconnect timer	The call was still in vector processing when the vector disconnect timer expired. The call dropped.
7	vec_act.c (07)	There is a mismatch between Attendant Vectoring and Call Vectoring between the VDN and the vector.
8	vec_act.c (08)	There is a mismatch between Attendant Vectoring and Call Vectoring between the incoming call and the VDN.
9	vec_act.c (09)	There is a mismatch between Attendant Vectoring and Call Vectoring between the incoming call and the vector.
10	Retrying announcement	During an **`announcement`** step, a **`collect digits`** step that contains an announcement, or a **`disconnect`** step, the announcement was not available, and the announcement queue (if specified) was full. The step is retried at regular intervals.

Summary of events (continued)

Event type	Event description	Explanation
11	No announcement available	During an **announcement** step, a **collect digits** step that contains an announcement, or a **disconnect** step, the announcement was not available for one of the following reasons: ● Announcement was not recorded ● Analog announcement was busied out ● Integrated announcement board was not installed ● Integrated announcement ports were busied out ● Integrated announcement was being recorded or restored
20	Call cannot be queued	A **queue-to split**, **messaging split**, or **check split** command failed to queue the call. **NOTE:** Event types 520, 521, 522 and 541 may be observed for the same call at the same time.
21	Queued to three splits	The call attempted to queue to four splits. Multiple split queuing allows the call to queue to a maximum of three splits simultaneously. If the call queued to one or more splits, and if it should now be dequeued from those splits and then queued elsewhere, one solution is to route the call to a station (which may be administered without hardware). Once this happens, the call is forwarded to the VDN that controls the next stage of the call.
22	vec_q_attd ( ), vect_q_atg ( ), vec_q_hunt ( ) (22)	Applies to Attendant Vectoring and indicates that the call is in the attendant queue and another attempt is made to queue the call to an attendant or hunt group, or the call is in the hunt group queue and an attempt is made to queue it to an attendant or too many attempts are made at queueing to the hunt group.
30	No TTR available	A **collect digits** command failed because: ● TN744 port was not available ● All queue slots were occupied
31	Dial-ahead discarded	Previously entered dial-ahead digits have been discarded via access of a(n) **adjunct routing**, **converse-on**, **route-to number**, or **messaging split** step.

Summary of events (continued)

Event type	Event description	Explanation
32	Prompting buffer overflow	The prompting digit buffer already contained the maximum of 24 digits when additional dial-ahead digits were entered by the caller. These additional digits are not stored.
33	ced digits left behind	A **`collect ced digits`** step collected digits from a UEC IE, and more than 16 digits were sent from the network.
34	cdpd digits left behind	A **`collect cdpd digits`** step collected digits from a UEC IE, and more than 16 digits were sent from the network
35	ced digits not available	A **`collect ced digits`** step collected digits from a UEC IE, and no digits were sent from the network, or no digits were present in the UEC IE.
36	cdpd digits not available	A **`collect cdpd digits`** step collected digits from a UEC IE, and no digits were sent from the network, or no digits were present in the UEC IE.
40	Messaging step failed	A messaging step failed because the Messaging Adjunct was not available. **NOTE:** Event types 540 and 541 may be observed for the same call at the same time.
50	Route -to step failed	A **`route-to`** step failed to reach the intended destination. **NOTE:** Event types 51 and 52 may provide more specific information regarding the reason for the failure. See Appendix I: Operation details for the route-to command on page 579.
51	No digits to route-to	The **`route-to digits`** step was unable to route the call because the previous **`collect digits`** step failed to collect any digits. This could result from an error in vector programming (for example, a **`route-to digits`** step appears without a preceding **`collect digits`** step). More often, however, this results because the caller was unable to enter the required digits (that is, the caller was using a rotary telephone), or because the caller was not provided with enough information to do so (as can be the case for auto-attendant applications).
52	No available trunks	A **`route-to`** command was unable to reach the specified off-switch destination due to a lack of available trunks.
53	Route-to step failed	The step was unable to seize a trunk because of a hardware problem or glare.

Summary of events (continued)

Event type	Event description	Explanation
54	LAI retry	Look Ahead Interflow **`route-to`** step failed because of glare. The route will be retried once.
55	Double coverage attempt	Coverage option on route-to step was ignored because double coverage is not allowed. This may happen when the call has covered to a VDN.
60	Adjunct route failed	An adjunct route failed for one of reasons indicated in event types 61 through 66.
61	Invalid destination	The **`adjunct routing`** command returned digits that did not represent a valid destination.
62	Adjunct route cancelled	The **`adjunct routing`** step was cancelled because another "routing" step (such as a **`queue-to split`** step) was encountered in the vector.
63	Queue before route	The **`adjunct routing`** command was skipped because the call had already been queued via a **`queue-to split`** or a **`check split`** command.
64	Adjunct link error	The **`adjunct routing`** command was cancelled for one of the following reasons: ● Link to the adjunct was down ● ASAI protocol violation prevented the call from completing ● Software resources to complete the call were unavailable
65	Agent not logged in	A Direct Agent Call was made to an agent who was not logged into the relevant split. Used for adjunct routing request only.
66	Agent not member of split	A Direct Agent Call was made to an agent who is not a member of the relevant split. Used for adjunct routing request only.
67	Invalid direct agent	A Direct Agent Call was made to an agent extension that is not valid. Used for adjunct routing request only.
70	Busy step for CO trunk	A CO trunk call reached a **`busy`** step in a vector without having previously received answer supervision. As a result, the caller continues to hear ringback rather than the busy tone.
80	Time not set	A **`goto`** step with a **`time-of-day`** conditional was processed, but the switch time was not set.

Summary of events (continued)

Event type	Event description	Explanation
81	No digits collected	No digits were collected and a comparison was requested against a digit string or in-table. The comparison test was considered false and the next step in the vector was executed.
90	Wait step music failed	A **`wait-time`** step with music was accessed, but the music was not connected. Music may not be administered correctly.
91	Wait step ringback failed	A **`wait-time`** step with ringback was accessed, but the ringback was not connected.
100	Redirect unanswered call	The call was sent to an agent via a vector, but, due to the Redirection on No Answer (RONA) feature, the call was redirected from the ringing agent.
101	Redirect of call failed	The call was sent to an agent via a vector, but, due to the Redirection on No Answer (RONA) feature, the call was redirected from the ringing agent. The call could not be redirected.
110	Converse no ANI digits	On a **`converse-on`** step with passing type **`ani`**, no information was available to populate the field.
111	Converse no qpos digits	On a **`converse-on`** step with passing type **`qpos`**, no information was available to populate the field.
112	Converse no prompt digits	On a **`converse-on`** step with passing type **`digits`**, no information was available to populate the field.
113	Converse drop during data	On a **`converse-on`** step, the converse agent hung up while data was being passed. This may indicate a port failure.
115	ASAI transfer converse	ASAI attempted a transfer of a call that was active at a **`converse`** step. The transfer failed, and vector processing continued at the next vector step.
116	Converse transfer denied	A transfer of a call that was active at a **`converse-on`** step was attempted. The transfer either failed or was denied, and vector processing continued at the next vector step.
117	Agent drops converse	While active on a **`converse-on`** step, an agent became available in a split associated with a **`queue-to split`** or **`check split`** step. The call was delivered to the nonconverse agent, and the converse agent was dropped.
125	Data return no digits	On a **`converse-on`** step, the converse agent activated data return but did not return any digits.

Summary of events (continued)

Event type	Event description	Explanation
126	Data return timeout	On a **`converse-on`** step, the converse agent activated data return but timed out while waiting to return digits. Vector processing continued at the next vector step.
140	Coverage conference denied	Coverage to a VDN in a coverage path was denied because more than one party was active on the call.
150	Invalid EAS hunt group used in the vector step	Either the skill hunt group was removed or the skill hunt group became a non-ACD hunt group.
151	Skill indirection used improperly	Either no VDN skills are administered or the vector command has skill indirection and EAS is not enabled.
160	No vector steps, ANI sent	ANI was sent to the CMS for a call that reached a VDN that accessed a vector with no steps defined.
170	ASA - invalid VDN	A **`check`** or **`goto`** test requested a comparison of ASA for a VDN that had been removed since the vector was programmed. The comparison test was considered false and the next step in the vector was executed.
200	ANI not avail - digits	A **`goto`** test requested a comparison of ANI against a digit string and ANI was not available for the call. The comparison test was considered false and the next step in the vector was executed.
210	Routing table not assigned	A **`goto`** test requested a comparison with a vector routing table that is not assigned or had been removed since the vector was programmed. The comparison test was considered false and the next step in the vector was executed.
211	No entries in routing table	A **`goto`** test requested a comparison with a vector routing table that has no entries. This is considered as a non-match.
212	ANI not avail - table	A **`goto`** test requested a comparison of ANI against “in-table” and ANI was not available for the call. The comparison test was considered false and the next step in the vector was executed.
220	EWT call not queued	A **`goto`** test for a call or converse data passing requested EWT for a call not in queue. In this case, the wait time was assumed to be infinite and the comparison was based on EWT > largest possible threshold.

Summary of events (continued)

Event type	Event description	Explanation
221	EWT not sent to VRU	The EWT “wait” time for the call was not sent to the VRU for a **`converse-on passing wait`** vector step because the call was not queued or the splits/skills that the call was queued to were unstaffed.
222	System clock change	The system clock was changed, therefore any calculations involving time (i.e., ASA and EWT) will be inaccurate.
230	II-digits not avail - digits	A **`goto`** test requested a comparison of II-digits against a digit string and II-digits were not available for the call. The comparison test was considered false and the next step in the vector was executed.
231	II-digits not avail - table	A **`goto`** test requested a comparison if II-digits against in-table and II-digits were not available for the call. The comparison test was considered false and the next step in the vector was executed.
240	No agent strategy found in VDN	The active VDN for the call, as determined by VDN override, did not have a BSR Available Agent Strategy.
251	Call is not incoming ISDN	Occurs when a **`reply-best`** command in a status poll vector receives and tries to process a non-ISDN call. Processing in the status poll vector terminated is without a reply being sent.
261	No “best” location found	A **`queue-to best`**, **`check-best`**, or **`reply-best`** command failed because the call vector was unable to calculate a best value or because no local best existed. Vector processing continues at the next step. Vectors in multi-site BSR applications won’t attempt to interflow calls in this situation.
262	Look-Ahead Interflow attempt failed	Interflow of the call failed: no trunk was available, LAI denial, or some other problem. Vector processing continues at the next step. In BSR applications, polling of this resource is temporarily suppressed.
271	No BSR app num in VDN	A **`queue-to best`**, **`check-best`**, or **`consider location`** command failed because the active VDN for the call as determined by VDN override has no BSR application number assigned. Processing continues with the next vector step. Only occurs in multi-site BSR applications.

Summary of events (continued)

Event type	Event description	Explanation
272	No BSR application plan administered	A **`queue-to best`**, **`check best`**, or **`consider location`** command failed because the application number assigned to the active VDN does not have an application plan assigned. Processing continues at the next step.
273	Location not on BSR form	A **`consider`** command failed because it refers to a location number that is not in the BSR Application form assigned to the active VDN. Vector processing continues at the next step.
274	Status Poll VDN field is blank	A **`consider`** command failed because the entry for this location on the BSR Application form does not contain a routing number for the status poll VDN.
275	Interflow VDN field is blank	A **`queue-to best`** or **`check-best`** command failed because the entry on the BSR Application form for the relevant location does not contain a routing number for the interflow VDN.
276	Agent Status Info Invalid	A **`consider location`** command failed because the status poll returned invalid data for an available agent (AIT, skill level, or occupancy is missing or out of range). Vector processing continues at the next step. Polling of this location is temporarily suppressed.
277	BSR Status Info Invalid	A **`consider location`** command failed because the status poll returned invalid EWT data. Vector processing continues at the next step. Polling of this location is temporarily suppressed.
278	No BSR Data in Response	A **`consider location`** command failed because the status poll did not return data in the DISCONNECT message. Vector processing continues at the next step. Polling of this location is temporarily suppressed.
279	No response from status poll	A **`consider location`** command failed because the status poll did not respond within the time allowed or because the status poll could not be performed. Vector processing continues at the next step. Polling of this location is temporarily suppressed.
280	Bad resp from status poll	A **`consider location`** command failed because it received an invalid response from the status poll such as an LAI acceptance message (such as ALERT or CONNECT). Vector processing continues at the next step. Polling of this location is temporarily suppressed.

Summary of events (continued)

Event type	Event description	Explanation
281	BSR EWT is infinite	A **`consider`** command failed because the EWT for the referenced split or skill is infinite. This may be because all agents are logged out or in AUX work, or because no queue slots are available. Vector processing continues at the next step. Polling of this location is temporarily suppressed.
282	BSR status poll attempt failed	A **`consider location`** command failed because the status poll attempt failed. See other events for specific reason. Vector processing continues at the next step. Polling of this location is temporarily suppressed.
283	BSR poll no trunks	A **`consider location`** command failed because there were no available trunks. Vector processing continues at the next step. Polling of this location is temporarily suppressed.
284	BSR poll seize fail	A **`consider location`** command failed because the status poll was unable to connect to a trunk due to a hardware problem. Vector processing continues at the next step. Polling of this location is temporarily suppressed.
285	BSR poll glare retry	The first status poll attempt for a **`consider location`** command was unable to connect to a trunk due to a race condition (the same trunk being seized for the outgoing call had an incoming call from the remote end). This status poll will be attempted once more. A second attempt failure will result in event 282.
287	Invalid status polling destination	An attempt was made to perform BSR polling over ISDN without B-Channel over a tandem trunk configuration that combines QSIG TSCs and AT&T TSCs (this type of interworking is not supported by Avaya's ISDN protocol).
291	No AITCI storage left	The network does not support the transport of all user data, so some user data was not sent. You can prioritize the user data using the Shared UUI Feature Priorities page of the ISDN Trunk Form. For more information, see Information Forwarding on page 151.
292	Data dropped by other app	
293	No room for reply-best information	The network or shared trunk setting does not support the transport of all data for the best resource. This is unlikely under normal circumstances since only 12 bytes of user information are required. Also see event 298.

Summary of events (continued)

Event type	Event description	Explanation
294	No room for in-VDN time	The network does not support the transport of all user data. You can prioritize the user data using the Shared UUI Feature Priorities page of the ISDN Trunk Form. For more information, see Information Forwarding on page 151.
295	No room for collected dgt	
296	No room for VDN Name	
297	No room for Other LAI	
298	Reply-best got bumped	The network or shared trunk setting does not support does not support the transport of all data about the best resource. (No other applications share user data included in a DISCONNECT message.)
299	In-VDN time got bumped	The network does not support the transport of all user data. You can prioritize the user data using the Shared UUI Feature Priorities page of the ISDN Trunk Form. For more information, see Information Forwarding on page 151.
300	Collected dgts got bumped	
301	VDN Name got bumped	
302	Other LAI got bumped	
303	Block: send reply-best	The transport of the best data for a **`reply-best`** command was denied because the trunk group is neither Supplementary Service b or Shared UUI.
304	No enhanced info is sent	During the execution of a **`queue-to best`** or **`check best`** step, information forwarding transport over this trunk was denied because the trunk group is neither Supplementary Service b nor Shared UUI. This event is not logged for LAI (for example, in execution of a **`route-to`** step) in order to permit backward compatibility. For more information, see Unexpected feature operations on page 499 as well as Information Forwarding on page 151 and Appendix E: Advanced multi-site routing on page 531.
310	NCR: Invoke trunk not ISDN	Check that only ISDN trunks are executing the vector steps where NCR is being invoked.
311	NCR: Bad NCR trunk admin	Check that all Trunk Form and Signaling Group form fields related to the NCR feature are correct.
312	NCR: No NCT PSTN service	Check that the PSTN service provider has activated the NCT feature for the ISDN trunk being used for NCT call redirections.

Summary of events (continued)

Event type	Event description	Explanation
313	NCR: No NCT outgoing trk	Check that the ISDN trunk group is administered as a two-way trunk group and that the Usage Allocation settings for the trunk have been set up correctly.
314	NCR: NCT outgo trk drop	Shows that the second leg of the NCT call has been dropped due to a trunk hardware problem, or that a vector step has been executed that returned and ISDN DISCONNECT message (such as a **`busy`** vector step).
315	NCR: PSTN NCT invoke err	The PSTN switch has not accepted the NCT invocation attempt. Check that the PSTN network switch complies with the NCT standards.
316	NCR: PSTN NCT netwrk err	The PSTN switch has accepted the NCT invocation attempt, but has rejected it due to some error condition within the network switch. Check that the `Network Call Redir` field on the Trunk form is administered correctly. Make a request to the PSTN service provider for troubleshooting assistance.
317	NCR: Used NCT trk-to-trk	NCT has not been successfully invoked, but the incoming call is still active as a switch trunk-to-trunk connection (this is only an informational message).
318	NCR: No NCD PSTN service	Check that the PSTN service provider has activated the NCD feature for the ISDN trunk being used for NCD call redirections.
319	NCR: NCD invalid PSTN nmbr	The PSTN switch has detected that the number used for the NCR invocation that was administered in the ~*r* **`route to number`** vector step or in the BSR Application Table's `VDN Interflow Number` field is an invalid PSTN number (the correct PSTN number used through switch administration).
320	NCR: NCD call connect err	The vector step has been executed before the vector step invoking NCD that sends an ISDN CONNECT message to the PSTN.
321	NCR: PSTN NCD invoke err	The PSTN has not accepted the NCD invocation attempt. Check that the PSTN network switch complies with the NCD standards. Make a request to the PSTN service provider for troubleshooting assistance.
322	NCR: PSTN NCD netwrk err	The PSTN switch has accepted the NCD invocation attempt, but has rejected it due to some error condition within the network switch. Make a request to the PSTN service provider for troubleshooting assistance.

Summary of events (continued)

Event type	Event description	Explanation
323	NCR: PSTN NCD max redirs	The PSTN has detected that the call has been redirected by NCD more that the public network “maximum number of call deflections limit” will allow. Modify vector processing to reduce the number of NCD attempts.
324	NCR: PSTN NCD no disc	The PSTN switch has not disconnected the ISDN trunk after performing the NCD or NCT call redirection. Make a request to the PSTN service provider for troubleshooting assistance.
325	NCR: Internal system err	The switch problem with call processing for the NCR invocation attempt. Alternately, for NCT, the first vector step at the redirected-to DEFINTIY endpoint is possibly not programmed with a call treatment vector step such as **`wait hearing ringback`**, **`wait hearing music`**, or **`announcement`**. Avoid the use of a vector step such as **`wait hearing silence`** or **`wait hearing i-silence`** for the first vector step at the redirected switch endpoint.
520	Split queue is full	A **`queue-to split`**, **`check split`**, or **`messaging split`** command was executed, but the call did not queue to the split because the queue (if administered) was full. To prevent this condition, use a **`goto step...if calls queued in split...>...`** before each **`queue-to split`** or **`check split`** step so that an alternative treatment may be provided for these cases.
521	Not vector-controlled	The split accessed by a **`queue-to split`** or **`check split`** command is not vector-controlled. As a result, the step is skipped.
522	AAS split cannot queue	A **`queue-to split`**, **`check split`**, or **`messaging split`** command was executed on an auto-available split (AAS), but the call did not queue to the split because all the agents were logged out by Redirection on No Answer (RONA).
540	AUDIX link down	AUDIX could not be accessed via a **`messaging split`** command, because the AUDIX link was down. As a result, the step is skipped.
541	Not a messaging split	The split administered for the **`messaging split`** command is not a messaging split (that is, it does not have a messaging type administered). As a result, the step is skipped.
542	Can’t connect idle agent	The call at the head of the queue can’t be connected to an idle agent.

Summary of events (continued)

Event type	Event description	Explanation
550	ASA - No staffed agents	A **`check`** or **`goto`** test requested a comparison of ASA for a split/skill that has no staffed agents. The comparison was based on ASA > largest possible threshold.
560	EWT no history for split	A **`goto`** test requested EWT for a split/skill that has not yet acquired history. The wait time in this case is assumed to be the default value.
561	EWT no split queue	A **`goto`** test requested EWT for a split/skill that has no queue. The wait time is assumed to be infinite. The comparison was based on EWT > largest possible threshold.
562	EWT split queue full	A **`goto`** test requested EWT for a split/skill whose queue is currently full. The wait time is assumed to be infinite. The comparison was based on EWT > largest possible threshold.
563	EWT split no working agents	A **`goto`** test requested EWT for a split/skill that has no agents logged in or all logged in agents are in the AUX work mode. The wait time in this case is assumed to be infinite and the comparison was based on EWT > largest possible threshold.
564	EWT split locked	A **`goto`** test requested EWT for a split/skill that is currently locked. The wait time is assumed to be infinite. The comparison was based on EWT > largest possible threshold.
565	EWT call no working agents	A **`goto`** test for a call or converse data passing “wait” requested EWT for a call that is queued only to splits/skills that have no agents logged in or that have all logged in agents in AUX work mode. In this case, the wait time was assumed to be infinite and the comparison was based on EWT > largest possible threshold.
2034	Denial event - BSR polling	A BSR polling over ISDN without B-Channel attempt has resulted in an illegal TSC interaction. Either an AT&T TSC was routed to a QSIG interface, or vice versa. The call is dropped and the denial event is logged.
	Denial event - BSR polling	A BSR polling over ISDN without B-Channel attempt has been denied for one of the following reasons: ● the terminated administered TSC endpoint is disabled ● The incoming nca-tsc call arrives at the wrong signaling group ● The max number of nca-tsc is set to 0.

Summary of events (continued)

Event type	Event description	Explanation
3201	Meet-Me Access chg TMO	The user changing the access code allowed the call to timeout to intercept treatment. The access code was not changed.
3202	Invld Num Digits MM Acc.	The user changing the access code entered too many digits. The access code was not changed.
3203	MM Extension not valid	The user changing the access code did not enter a valid extension.
3204	MM Access Chg Not a VDN	The user changing the access code entered a non Meet-me Conference VDN extension.
3205	MM Invalid Access Entered	The user changing the access code did not enter the correct access code. The access code was not changed.
3206	MM Access Obj/SAT Busy	An administrator is making changes to the Meet-me Conference VDN, so the user cannot change the access code using a feature access code. Try again later.
3207	Merge Meet-me Conf call	A user tried to access an existing Meet-me Conference call and was denied.
3208	Serv Observ Meet-me VDN	A user tried to service observe a Meet-me Conference call. This is not allowed.
3209	Meet-me Conf call full	A user tried to access a Meet-me Conference call that was already full.
3210	Wrong MM Acc. code dialed	A user trying to access a Meet-me Conference call dialed the wrong access code.
3211	Chg Station no Cons/Perm	The station attempting to change the access code does not have console permissions COS.
3212	VDN not a meetme type	The VDN that was called is not a Meet-me Conference VDN.
3213	MM Invalid Conf Ctrlr Sta	If controlling extension is filled in and the station and controller do not match.
3214	MM Inv Trk not Remote Acc	The trunk used to access the Meet-me Conference is not a remote access trunk.
3215	MM Invalid Station Type	If controlling extension is blank and the station type is invalid (for example, and attendant console).
3216	Conf/Transfer 2 Meet-me	A user cannot conference or transfer another call into a Meet-me Conference call.

Clearing events

When you have finished your review of the event log, you can remove events from the error log. You must be use superuser login ID to clear events.

To clear events from the error log, enter **`clear events`** at the command prompt and press **ENTER**. This command clears all events from the event buffer space within the error log. It does not delete any other entries in the error log.

Appendix E: Advanced multi-site routing

This appendix supplements the Look-Ahead Interflow (LAI) and Best Service Routing (BSR) chapters.

This appendix is intended for users whose call center networks meet either or both of the following criteria:

- Five or more switches in the network
- Combination of low- and high-volume locations

This appendix includes the following topics:

Application architecture in multi-site BSR

Multi-site applications may be structured in a variety of ways. In general, however, most applications will fit one of two models: distributed or centralized. When each switch in a network may interflow calls to other switches and receive interflows, this is called a distributed system. A centralized system, by contrast, is one in which all calls are initially delivered to a single call center (the “hub”) and distributed from this site to queues at remote switches. A centralized system requires greater inter-switch trunking, since a greater percentage of calls need to be redirected. However, it may be an appropriate configuration if your organization has a significant investment in VRU and CTI technology at the hub.

Which architecture you choose for an application has direct implications for your choice of user adjustments and polling patterns.

User adjustments

User adjustments in **`consider split`** and **`consider skill`** steps may be set at the user's discretion. In distributed multi-site applications, however, adjustments must be carefully considered because of their potential affect on costs and inter-switch trunk capacity. In centralized applications all calls are redirected anyway so it's OK to use adjustments of "0". In distributed applications, though, a user adjustment of "0" for a **`consider location`** step is almost never practical or efficient.

In distributed applications, the smaller the adjustment the closer the load balance across the network, but the greater the percentage of calls redirected between switches (and thus the greater the demands on inter-switch trunking). Higher adjustments reduce interflows, but at the cost of allowing greater imbalance in the load between switches. It will take some time and effort to find the best combination of user adjustments in any particular network, but Recommended initial user adjustments on page 532 contains recommended ranges for initial user adjustments under different conditions. Adjustments may vary between different call center applications so apply these guidelines for each of your applications separately.

Recommended initial user adjustments

Recommended adjustments…	If the following criteria apply…
10–15	• You want to balance wait times across the network as much as possible. • Trunk facilities between switches are plentiful. • Each switch receives more than 1 call every 10–15 seconds (more than 240–360 calls/hour) for this application.
20	• Balancing wait times across the network is important to you. • Adequate trunk facilities are available to support the desired balance. • Each switch receives more than 1 call every 20 seconds (more than180 calls/hour) for this application.
30 or higher	• Gains in agent efficiency are more important to you than balancing wait times across the network. • Trunk facilities are scarce. • Call interflow is costly. • Each switch receives no more than 1 call every 30 seconds (around 120 calls/hour or lower) for this application.

In your first multi-site application, it is recommended that you begin with a remote adjustment of 30. This can easily be reduced later if inter-switch trunking is under-utilized. On the other hand, if trunk exhaustion is a common occurrence then user adjustments are probably set too low. Care should be taken not to lower remote user adjustments to such an extent that all trunk resources are regularly exhausted. When trunks are exhausted, no further load balancing can take place and the overall balance may deteriorate.

User adjustments should also be set high enough that calls are not interflowed to gain the equivalent of a fraction of a queue position. The following equation will give you the minimum recommended user adjustment for each remote switch:

$$\frac{\mathrm{AverageCallHandlingTime}}{\mathrm{NumberOfFullTimeEquivalentAgents}} \leq \mathrm{UserAdjustment}$$

Adjustments for remote locations will probably be in the range of 10–30 in most distributed applications.

Is there a reliable relationship between user adjustments and the balance in wait times across a network?

Changing conditions can produce significant variations in such a balance, but on average you can predict the balance in wait times for a given user adjustment.

Let’s say a user adjustment of 20 is chosen for all remote resources in a network and all the remote sites are polled. When waiting times are short (< 100 secs), the highest and lowest EWTs for this application on the network should stay within a range of approximately 20 seconds (30–50 seconds, for example). When waiting times are long (> 100 secs), the highest and lowest EWTs for the application should stay within a range of approximately 20% (5 to 6 minutes, for example).

Status polling in BSR

Status polls are the key element in multi-site BSR applications. Status polls provides the communication links between a switch that wants to interflow a call and the switches that might service that call.

The vectors you write in multi-site applications must balance the costs of time and trunk usage with the benefit of better customer service. BSR is designed to help you achieve this balance, incorporating mechanisms to maximize improvements in customer service while minimizing inter-switch communications with its attendant delays and trunk usage. This section explains those mechanisms and the benefits they provide as you write vectors.

How long do status polls take?

One **`consider location`** step polls one remote location. Does this mean that an optimal multi-site BSR application polls every switch in a network? No.

Let's look at an example of a moderately large network, containing 16 switches. The primary vector on switch #1 could be written as shown in the following vector example. Polling response times are variable. Let's assume that this is a slow response network and that each status poll takes 1 second. The consider series in this vector could add as much as 15 seconds to a call's time in vector processing! In fact, the vector shown below is provided as an example of what NOT to do. The benefits of BSR can be obtained much more efficiently.

Intelligent polling for multi-switch networks

```
1.   wait time 0 secs hearing ringback
2.   consider skill 1 pri m    adjust-by  0
3.   consider skill  2 pri m   adjust-by 20
4.   goto step 20 if expected-wait for best = 0
5.   consider location 1       adjust-by 30
6.   consider location 2       adjust-by 30
7.   consider location 3       adjust-by 30
8.   consider location 4       adjust-by 30
9.   consider location 5       adjust-by 30
10.  consider location 6       adjust-by 30
11.  consider location 7       adjust-by 30
12.  consider location 8       adjust-by 30
13.  consider location 9       adjust-by 30
14.  consider location 10      adjust-by 30
15.  consider location 11      adjust-by 30
16.  consider location 12      adjust-by 30
17.  consider location 13      adjust-by 30
18.  consider location 14      adjust-by 30
19.  consider location 15      adjust-by 30
20.  queue-to best
21.  announcement 1001
22.  wait time 60 secs hearing music
23.  goto step 21 if unconditionally
```

First, even in very large networks you can obtain nearly all of the possible benefits in agent utilization with very few polling connections. In a network of 16 switches, 99% of the total benefits possible with BSR can be obtained if each switch polls just 4 others. See How many switches should one switch poll? on page 537 for more information on this topic.

Now our vector looks like the following. Is polling time now cut from 15 seconds to 4 seconds, proportional to the reduction in **`consider`** steps?

```
1.   wait time 0 secs hearing ringback
2.   consider skill 1 pri m    adjust-by  0
3.   consider skill  2 pri m   adjust-by  0
4.   goto step 9 if expected-wait for call = 0
5.   consider location 5       adjust-by 30
6.   consider location 10      adjust-by 30
7.   consider location 13      adjust-by 30
8.   consider location 15      adjust-by 30
9.   queue-to best
10.  announcement 1001
11.  wait time 60 secs hearing music
12.  goto step 10 if unconditionally
```

In fact, polling time in this vector may be around 0.4 seconds per call because of mechanisms in BSR that constantly react to network conditions and resource usage to minimize the number of status polls. These mechanisms, whose combined operation is called “intelligent polling,” also function to make each status poll as productive as possible.

Intelligent polling

A BSR application will only poll the switches that are likely to provide the best service at any given time. If a remote switch is polled and returns an adjusted EWT greater than that of the current best resource, polling of the remote switch will be suppressed for a period of time proportional to the difference between the two adjusted EWT values. (In other words, polling of a given location is suppressed whenever the adjusted EWT returned by that location is subsequently replaced by a better adjusted EWT from another resource.) The **`consider`** step for this location will be skipped during this period and vector processing will continue at the next step. When the suppression period is over, the **`consider`** step will once again poll this location. If the location returns the best adjusted EWT, the next call processed by the vector will also cause this location to be polled. If it is not the best, polling will again be temporarily suppressed, and so on.

If no calls are in queue at the remote location an agent might become available at any moment, and thus BSR will never suppress polling for longer than 5 seconds in such situations. BSR will never suppress polling of any remote location for more than 60 seconds, regardless of the differences between adjusted EWT returned by different switches.

Other conditions can also suppress status polls to a location:

- resource exhaustion (no trunks available, queue full)
- administration errors (badly written vectors, or no application plan)

This feature significantly reduces the average number of status polls placed per call. The greater the call volume, the greater the percentage reduction. Let's take another look at the vector in Screen 2.

Let's assume that the network is operating in a balanced state. EWTs are 30 seconds at all locations, and a call arrives every 3 seconds at each site. Adjusted EWTs are 30 seconds at the origin switch and 60 seconds for each remote switch. After each status poll under these conditions, polling will be suppressed for 30 seconds. Each remote location is polled therefore, by every 10th call. On average, this means that each call polls any one location 0.1 times. Since there are four **`consider`** steps, each call makes 0.4 polls. Remembering the 1-second polling response time given at the beginning of the example, the average time added to call processing for each call is 0.4 seconds.

The 1st-found available agent strategy, discussed in Best Service Routing (BSR) on page 229, can cut average polling times further. With the 1st-found strategy, BSR will skip all subsequent **`consider`** steps in a series if a resource with an available agent is found and deliver the call to that resource.

Efficient polling patterns in large networks

Unless you have a small network, you won't benefit by having every switch poll every other switch. This section explains how many remote locations each switch needs to poll, and it provides guidelines for selecting which locations any given switch should poll.

How many switches should one switch poll?

It's not necessary to poll every switch in larger networks. Because of BSR's intelligent polling capabilities, you can obtain 99% of the possible benefits in agent utilization with very few polling connections.

For an example, let's look at a laboratory network of 16 switches that is used for simulations of BSR multi-site applications. As shown in the following table, approximately 99% of the possible benefits were obtained when any one switch polled 4 others.

Effectiveness of status polls in a 16-switch network

Number of remote sites polled by each switch	ASA across the network (seconds)	Approximate percentage of total benefits obtained
0	192.8	0%
1	26.2	89%
2	10.6	95%
3	7.6	98%
4	6.5	99%
15	4.7	100%

For each switch to poll the other 11 switches in the network would only produce an additional 1% gain in ASA and agent utilization—an improvement which would be more than offset by the cost of additional messaging and trunking.

In most situations, you'll obtain the optimal results with your multi-site BSR applications if you follow the polling guidelines shown in the following table.

Recommended number of locations to poll

If there are this many switches in the network…	Each switch should poll…
2–4	all the other switches
5–10	3 other switches
11–20	4 other switches
21–40	5 other switches
41 or more	6 other switches

Which remote switches should each switch poll?

In networks with fewer than 5 switches, each switch can productively poll all the other switches in the network. In larger networks, each switch need not poll every other switch. But which switches should each switch poll? We'll use the term "polling patterns" to describe the relationships between switches in multi-site BSR applications.

Here are two patterns to avoid. They're simple and seem intuitively obvious, but they don't usually yield the best possible results:

- Mutual polling: As much as possible, 2 switches shouldn't poll each other. This is unavoidable in small networks, but in large networks it can and should be minimized.
- Polling chains: For example, if switch A polls B & C, B polls C & D, and so on, this is a polling chain.

You may want to experiment with polling patterns appropriate to your own network and applications (if you're not constrained by the physical structure of your network). The following table provides a template for creating polling patterns for applications of up to 12 switches. In the majority of situations, these patterns will produce results that are close to optimal. To use this table, first assign a number from 1 to x to each switch in your application. Next, find the column that matches the number of switches in your application.

As you read down that column, you'll see which switches each particular switch in the application should poll.

Polling patterns for networks of 5–12 switches

This switch ...	Should poll the specific switches shown in the column for your network size							
	5	**6**	**7**	**8**	**9**	**10**	**11**	**12**
1	2,4,5	2,4,5	2,4,6	2,4,7	2,4,6	2,4,7	2,4,8,10	2,4,8,9
2	3,5,1	3,5,6	3,5,7	3,5,8	3,5,7	3,5,8	3,5,9,11	3,5,9,10
3	4,1,2	4,6,1	4,6,1	4,6,1	4,6,8	4,6,9	4,6,10,1	4,6,10,11
4	5,2,3	5,1,2	5,7,2	5,7,2	5,7,9	5,7,10	5,7,11,2	5,7,11,12
5	1,3,4	6,2,3	6,1,3	6,8,3	6,8,1	6,8,1	6,8,1,3	6,8,12,1
6		1,3,4	7,2,4	7,1,4	7,9,2	7,9,2	7,9,2,4	7,9,1,2
7			1,3,5	8,2,5	8,1,3	8,10,3	8,10,3,5	8,10,2,3
8				1,3,6	9,2,4	9,1,4	9,11,4,6	9,11,3,4
9					1,3,5	10,2,5	10,1,5,7	10,12,4, 5
10						1,3,6	11,2,6,8	11,1,5,6
11							1,3,7,9	12,2,6,7
12								1,3,7,8

In applications of more than 12 switches, the following table provides the formulae you need to figure out the optimal polling pattern.

Polling pattern formula for large networks

Number of switches in application	Switch "i" should poll...
13 or 16	i + 1, i + 3, i + 7, i +11
14 or 19	i + 1, i + 3, i + 7, i + 9
15	i + 1, i + 3, i + 7, i + 10
17 or 20	i + 1, i + 3, i + 7, i + 12
18	i + 1, i + 3, i + 7, i + 13

Polling pattern formula for large networks (continued)

Number of switches in application	Switch “i” should poll…
21–23	i + 1, i + 3, i + 7, i + 15, i + 17
24	i + 1, i + 3, i + 7, i + 15, i + 19
25	i + 1, i + 3, i + 7, i + 15, i + 20

To use one of these formulae, first assign a number from 1 to x to each switch in your application. Then, in the left-hand column of the table, find the number of switches in your application. The corresponding formula in the right-hand column is the one you should use.

In the formulae, “i” is the number of the switch for which you’re calculating a polling pattern. For example, let’s say you want to calculate the polling patterns in an application with 16 switches. The formula to use is

i + 1, i + 3, i + 7, i + 11

as shown in the first row of the table. Here are the actual results of this formulae for the first 5 switches in this 16-switch application.Notice that the numbers “wrap” (start over at 1) after you’ve polled the last switch in the network: switch 5 polls switch 16 as its fourth poll, and then the polling pattern for switch 6 has switch 1 in the fourth position.

Switch number…	Should poll switches…
1	2 ,4, 8, 12
2	3, 5, 9, 13
3	4, 6 ,10, 14
4	5, 7, 11, 15
5	6, 8, 12, 16
6	7, 9, 13, 1
7	8, 10, 14, 2

Considerations for low volume splits/skills

Very small resources (for example, 2–3 agents) have special needs. With BSR, it is easy to obtain a very close balance of wait times across a network of call centers. however, for very small splits/skills, wait times for each call can vary significantly.

To see why this is, let's take an extreme example of a split with a single agent logged in with one call active and none in queue. Average call handling time is 3 minutes. Now, if a new call arrives in queue, that call could be answered almost immediately—or it might wait for 3 minutes or more. The variation in wait times is perhaps 5–180 seconds.

In general, the fewer agents logged into a split/skill, the greater the variability in wait times because agents become available less often. BSR will naturally favor large resources, steering calls away from smaller resources when there are no available agents or wait times are not the best in the application. This tendency helps reduce the possibility that an individual caller might have a disproportionately long wait at a small resource.

If your network includes very small splits/skills, you have three options:

- If your operation is not badly affected by a small percentage of calls having variable wait times, simply use BSR normally across the network.
- If your principal concern is that a call does not wait in queue while an agent is available elsewhere, use BSR normally but write primary vectors at smaller locations to perform rapid look-ahead attempts to other resources once the call has been queued. (Rapid LAI vector loops use the **`interflow-qpos`** conditional, which is an enhancement to LAI. For more information on LAI and the **`interflow-qpos`** conditional, see Look-Ahead Interflow (LAI) on page 203.) For an example of this type of vector, Using LAI as a backup on page 542.
- If you want to answer every caller quickly, then the following configuration is recommended. Do not deliver or queue calls directly to the very small resources. Deliver or queue all incoming calls to larger resources, and use BSR to balance the load across these larger locations. Some or all of the larger locations should then perform rapid look-ahead attempts to one or more of the smaller resources. In this way, the members of the very small resource become an extension of the agent pool at one of the larger call centers. For an example of this design, see Single-queue FIFO hybrid configuration on page 543.

In any network, avoid having several large resources poll or make look-ahead attempts to a very small resource. Since the status at the very small resource changes infrequently, frequent polls to that resource are wasteful. A very small resource should receive look-ahead attempts or be polled only by other small resources or by one large resource.

Minimizing variations in wait time

When a network contains (or when a call center application combines) large resources and very small resources, BSR and LAI can be effectively combined. This section presents two sample vectors. The first example shows a primary vector intended for the smaller resources in a network when you want to avoid having a call in queue at one call center while an agent is available at another. This design will reduce wait time variation as well. The second example illustrates a primary vector for larger locations: this example shows you the best way to minimize wait times across a network

Using LAI as a backup

As noted above, if your principal concern is that a call not wait in queue while an agent is available elsewhere, use BSR at all locations in the network. At smaller locations, write primary vectors that will perform rapid LAI attempts to other (preferably larger) resources once the call has been queued.

```
1.  wait time 0  secs hearing ringback
2.  consider skill 1st pri m adjust-by  0
3.  consider location 12     adjust-by 30
4.  consider location 22     adjust-by 30
5.  goto step 7 if expected-wait for call < 600
6.  disconnect after announcement 3501 "Due to heavy call volume..."
7.  queue-to skill best
8.  announcement 3500 "Thanks for calling...."
9.  goto step 13 if expected-wait for call < 90
10. wait time 45 secs hearing music
11. announcement 3502 "Still busy..."
12. goto step 9 if unconditionally
13. route-to-number 913031234567 with cov n if interflow-qpos = 1
14. wait time 5 secs hearing music
15. goto step 13 if unconditionally
```

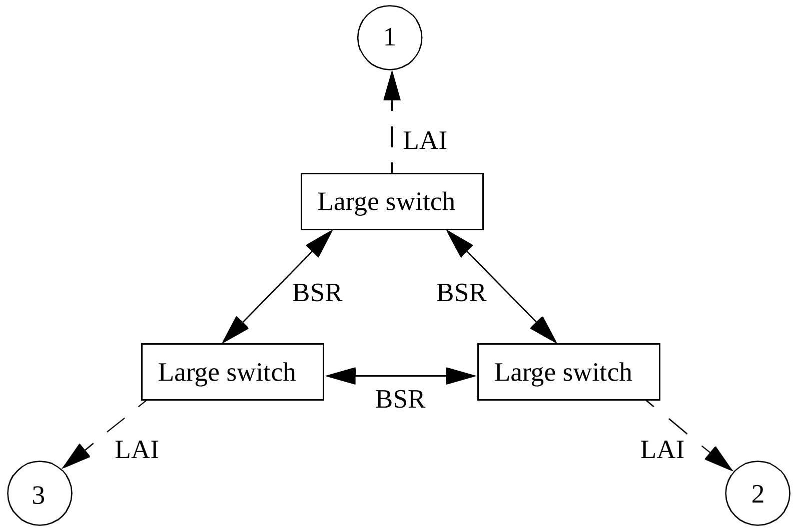
Steps 1 to 4 comprise a typical BSR vector. The origin switch considers a local resource and 2 remote resources. Before queuing or routing the call, however, the vector checks the expected wait time for the best resource. If this is 10 minutes or more, the caller receives a ‘busy” announcement. Otherwise, the queue-to best step sends the call to the best resource. Two vector loops follow: one 45-second loop with music and a delay announcement, and one 5-second loop that uses LAI. If the call is queued successfully in step 7 the first announcement loop (steps 9-12) executes until the call gets within a certain range of the head of the queue (at which point EWT is less than 90 seconds). At this time, step 9 sends the call to the second loop, where LAI attempts are placed every 5 seconds for the call at the head of the interflow eligible queue (**`interflow-qpos=1`**). If an agent becomes available at the larger remote resource, any call at the head of the eligible queue at the smaller location is outflowed to the larger resource, normally within a period of 5 seconds.

Single-queue FIFO hybrid configuration

To minimize variations in wait time across a network, the best strategy may be to let only the call centers with the larger resources receive calls. The following figure shows a network of 3 large and 3 small resources (call centers with large splits/skills and call centers very small splits/skills in the same application).

The large locations use BSR and all poll each other, while each location with a small resource (numbered 1, 2, 3) is treated as a satellite of one of the larger locations and only receives calls interflowed from that location. (Mutual polling is not optimal in larger networks, but it's OK for switches in such a small network to poll each other.) So BSR is used to balance the load between the locations with the larger resources. Then, each large switch executes a rapid LAI vector loop to one small switch to look for available agents. Since calls never queue at the small switches, the problem of highly variable wait times at the small resources is eliminated. This strategy will also give the best balance in wait times across resources.

Hybrid application architecture



The following vector example shows the primary vector that would be used at the large locations with this strategy. This vector is almost identical to the vector shown in Using LAI as a backup on page 542 above. The differences are at the application level. In contrast to the previous example:

- Only the locations with the larger resources receive calls.
- The primary vector shown here resides on the larger switches.

Steps 1 to 4 comprise a typical BSR vector. The origin switch considers a local resource and 2 remote resources. Before queuing or routing the call, however, the vector checks the expected wait time for the best resource. If this is 10 minutes or more, the caller receives a ‘busy” announcement. Otherwise, the queue-to best step sends the call to the best resource. Two vector loops follow: one 45-second loop with music and a delay announcement, and one 5-second loop that uses LAI. If the call is queued successfully in step 7, the first announcement loop (steps 9-12) executes until the call gets within a certain range of the head of the queue. At this time, step 9 sends the call to the second loop, where LAI attempts are placed every 5 seconds (only for the call at the head of the interflow eligible queue). If an agent becomes available at the smaller resource, any call at the head of the eligible queue at the larger location is outflowed to the smaller resource, normally within a period of 5 seconds.

Vector combining BSR and LAI

```
1.  wait time 0  secs hearing ringback
2.  consider skill 1st pri m adjust-by  0
3.  consider location 120    adjust-by 30
4.  consider location 220    adjust-by 30
5.  goto step 7 if expected-wait for best < 600
6.  disconnect after announcement 3501 "Due to heavy call volume..."
7.  queue-to skill best
8.  announcement 3500 "Thanks for calling...."
9.  goto step 13 if expected-wait for call < 90
10. wait time 45 secs hearing music
11. announcement 3502 "Still busy..."
12. goto step 9 if unconditionally
13. route-to-number 913031234567 with cov n if interflow-qpos = 1
14. wait time 5 secs hearing music
15. goto step 13 if unconditionally
```

Similar vector loops can be added to the interflow vectors at each of the large switches. In other words, each vector that processes calls at the larger locations can use rapid LAI loops to interflow calls to its satellite resource. This system maximizes agent utilization and the distribution of call load while evening out wait times across the network.

Appendix F: Advanced information forwarding

This appendix explains ISDN (BRI or PRI) trunk group setting interactions with Information Forwarding, UCID, and Multi-Site Routing.

User information included in the SETUP message for an outgoing call (at the sending switch) or DISCONNECT message sent back for an incoming call (at the receiving switch) is based on the trunk group settings at the sending or receiving sites.

The “shared” user information forwarding supports various data items (UCID, collected digits and In-VDN-Time) in addition to “shared forwarding” of LAI Info (VDN-Name and Other-LAI) and ASAI provided user data. “Shared forwarding” over non-QSIG ISDN trunks packs the data items in a codeset 0 UUI IE (called shared UUI), where each item consists of a two-byte header (application ID and data length). “Shared forwarding” over QSIG trunks transports the data items as Manufacturer Specific Information (MSI) in codeset 0 Facility IEs.

BSR and “shared data forwarding” (UCID and other new data items) requires QSIG or the shared `UUI IE Treatment` setting with non-QSIG trunk groups on both the sending (outgoing trunks) and receiving (incoming trunks) at the switch. Shared settings are also required on tandem trunk connections through the switch that routes these calls. LAI Info, UCID, collected digits, In-VDN-Time and ASAI provided user data can be forwarded with a call in the SETUP message (LAI or BSR interflowed call, a tandemed call, for UCID with any outgoing call and for ASAI user data any adjunct routed outgoing call). With the DEFINITY R6.3 capabilities, only BSR reply-best data is returned with a BSR poll call and only ASAI user data is returned for a non-poll call in a DISCONNECT message (both types of data will not be included in the same DISCONNECT message). Shared UUI Priority settings do not affect what is put in the DISCONNECT message or data forwarded over QSIG trunks.

The protocol (QSIG or non-QSIG) is set on page 1 of the ISDN trunk group form using the `Supplementary Service Protocol` field. QSIG type as defined for shared MSI is protocol type **b** (another protocol type “d”, ECMA QSIG is considered non-QSIG for Information Forwarding). The `Send Codeset 6/7 LAI` field on page 2 indicates whether or not to include an LAI IE in the SETUP message. The codeset used (6 or 7) is determined by the `Codeset to Send TCM, Lookahead` field on page 1. The `Send UCID` field on page 2 indicates whether or not the UCID data item should be included as user information with calls routed over this trunk group. The `Send Codeset 6/7 LAI IE` field is ignored for BSR polls over the trunk group (an LAI IE will never be included with BSR calls).

Non-QSIG protocol

`UUI IE Treatment` set to `service-provider` includes any application provided UUI in a codeset 0 UUI IE on a non-shared basis. That is, the data portion of the UUI IE only includes user info in the SETUP or DISCONNECT messages as provided by an application such as ASAI without the shared App-ID and length header fields. User data from only one application can be included in non-shared UUI. This setting would be used for non-QSIG trunk groups when service-provider functionality is wanted (for example, where shared forwarding of the new data items is not required or for trunk groups to other vendor switches or network services that need user information from the trunk group in a non-shared UUI IE such as provided by ASAI). Incoming calls received with shared user information (shared UUI IE) that are routed outgoing over a non-QSIG service-provider trunk group will forward only ASAI provided user data in a non-shared UUI IE.

`UUI IE Treatment` set to `shared` allows all applications to include data items in the UUI IE on a shared forwarding basis. The Shared UUI Feature Priorities page settings along with the `Max. Size of UUI Contents` field on page 2 and the features configured for the system determines what actually is included in the UUI IE. This is the normal setting for non-QSIG trunk groups that route calls to the switch over private or public networks when information forwarding is required and must be used for BSR.

QSIG trunk group

`UUI IE Treatment` set to `service-provider` forwarded ASAI provided user data in a non-shared codeset 0 UUI IE and all other user data in codeset 0 Facility IEs as MSI. In this case the `Max. Size of UUI Contents` field is not relevant and the Shared UUI Feature Priorities page does not show nor apply. This setting would only be used for QSIG trunk groups to pre-R6.3 DEFINITY switches for compatibility with existing ASAI applications or when service-provider functionality is wanted (e.g., where shared forwarding of the new data items is not required or for trunk groups to other vendor switches that need user information from the trunk group in a non-shared UUI IE such as provided by ASAI). Incoming calls received with shared data (shared UUI IE) routed out over a QSIG service-provider trunk group, will separate any ASAI provided user data included in the shared UUI IE and forward it in a non-shared UUI IE.

`UUI IE Treatment` set to `shared` will forward all user information including ASAI provided user data in codeset 0 Facility IEs as MSI in the SETUP or DISCONNECT message. The UUI IE is never included over a shared QSIG trunk group. In this case, the `Max. Size of UUI Contents` field and the Shared UUI Feature Priorities page do not apply. This is the normal setting for QSIG trunk groups to the switch when information forwarding is required and must be used for BSR.

“Send Codeset 6/7 LAI IE” option interactions

The `Send Codeset 6/7 LAI IE` option is independent of the `Supplementary Service Protocol` and `UUI IE Treatment` settings to allow additional flexibility. The switch can have a mix of trunk groups set with non-QSIG or QSIG protocol and with `service-provider` or `shared` settings. Calls interflowed over the shared non-QSIG trunk groups will contain the data items to be forwarded with the call in the UUI IE while calls interflowed over the non-QSIG service-provider trunk groups will not (except for ASAI which can always be sent in UUI). Calls interflowed over the QSIG trunk groups will always have MSI user information (except for ASAI whose transport method depends on the `UUI Treatment` setting).

When a call is LAI interflowed over a non-QSIG service-provider trunk group, the `Send Codeset 6/7 LAI IE` option being active will result in just the LAI IE being forwarded with the call in a SETUP message. When interflowed over a non-QSIG shared trunk group, setting the `Send Codeset 6/7 LAI IE` to yes includes a codeset 6/7 LAI IE in the SETUP message in addition to the same LAI information included as shared data in the UUI IE. If necessary and appropriate, the LAI information fields (and others) can be set to “blank” on the Priorities page to exclude these data items from the UUI IE. See Determining user information needs on page 158 for details. When interflowed over a QSIG service-provider or shared trunk group with `Send Codeset 6/7 LAI IE` active, the LAI information will be included as both MSI and in the LAI IE. However, in this case there is no mechanism to eliminate the duplication of data if the codeset 6/7 LAI IE is required.

These combinations can be used when calls are LAI interflowed to the switches previous to the switch with existing ASAI applications using ASAI provided UUI that may or may not be using the LAI IE. Note that codeset 6/7 IEs are not defined for QSIG and other vendor switch treatment of calls with a LAI IE is undefined (could be ignored, blocked, or misinterpreted).

When the trunk group is set to non-QSIG and shared or to QSIG (service-provider or shared), it is recommended that the `Send Codeset 6/7 LAI IE` option should not also be set to **y** due to the overhead of sending duplicate information. In some cases, this configuration could exceed the SETUP message and/or user information byte count limits for the network and result in the user information being dropped. Also, transport could cost more in networks which charge for user transport by quantity of bytes transported. An administration warning message will be given when this combination is set for the trunk group. In fact this combination is not recommended except in very limited cases where a mix of early and later switches can be reached over the same trunk group (via a public or switched private network) using Look-Ahead Interflow, and where BSR or UCID is not active or being used and the data that needs to be forwarded with the call can be limited to that supported by the network.

The `Send Codeset 6/7 LAI IE` option must not be set to **y** with trunk groups (or in switches) where calls will be interflowed over public networks or virtual private networks that do not support codeset 6/7 transport. In these cases, the codeset 6/7 IE will not be forwarded or the calls may not be routed by the network (blocked due to protocol errors). This can happen in some international situations, notably over networks in Germany.

Summary of what is included in the SETUP message

		Supplementary services protocol	
UUI IE Treatment	**Send Codeset 6/7 LAI IE**	**Non-QSIG (other than b)**	**QSIG (SS b)** [1]
service-provider	n	ASAI provided user info in codeset 0 UUI IE	ASAI provided user info in a codeset 0 UUI IE and all other user info in codeset 0 MSI
	y	ASAI provided user info in codeset 0 UUI IE & a codeset 6/7 LAI IE	ASAI provided user info in codeset 0 UUI IE, all other user info in codeset 0 MSI and a codeset 6/7 LAI IE [2]
shared	n	All user info in a shared codeset 0 UUI IE	All user info in codeset 0 MSI
	y	All user info in a shared codeset 0 UUI IE & a codeset 6/7 LAI IE [3]	All user info as codeset 0 MSI and a codeset 6/7 LAI IE [3]

1. MSI is sent in codeset 0 Facility IEs.

2. With this combination, the LAI information (LAI Name and Other LAI) will be sent both as MSI (in a Facility IE) and in the LAI IE. Note that LAI IE and shared MSI operation with other vendor switches is undefined.

3. With this combination, the LAI information (VDN-Name and Other-LAI) will be sent in both the UUI IE and in the LAI IE (setting the UUI Priorities for these items to “blank” can eliminate the duplication).

When to use specific trunk group options

Situation	Trunk group option settings: UUI IE treatment: Non-QSIG	QSIG	Send Codeset 6/7 LAI IE
Trunk groups over which information forwarding is not required (for LAI, BSR or UCID transport).	service-provider	service-provider	n
Non-LAI interflow or tandem calls to service providers or other vendor switches that do not recognize shared UUI.	service-provider	service-provider	n
LAI to pre-R6.3 switches over networks that block codeset 6/7 IE calls.	service-provider	service-provider	n
LAI to pre-R6.3 switches over networks that allow codeset 6/7 (traditional LAI) with or without ASAI applications that use UUI and/or LAI Info	service-provider	service-provider [1]	y
LAI over public/virtual private network to mixed R6.3 and earlier switches, where the Avaya switches have shared information forwarding. The pre-R6.3 switches may use LAI Info in an ASAI application, but must not use UUI.	shared [2]	shared [2]	y
LAI over public/virtual private network to mixed R6.3 and earlier switches. The R6.3 and earlier switches may use LAI info or UUI in an ASAI application.	service-provider [3]	service-provider [2]	y
BSR and/or LAI to all R6.3 or newer switches [4]	shared	shared	n

1. With this combination, the LAI information will be sent both as MSI (in Facility IEs) and in the LAI IE.
2. With this combination, the LAI information (LAI Name and Other LAI) will be sent in both the UUI IE and in the LAI IE.
3. The LAI IE and ASAI non-shared UUI is supported, but BSR, UCID and other new data items are not.
4. All switches interflowed to must be R6.3 or newer with “shared” incoming and outgoing trunk group settings. Tandeming/interflowing through R6.3 or later switches requires “shared” settings. Switches tandemed through can be older than R6.3 (or other vendor switches that pass codeset 0 UUI or MSI transparently). This is the only combination that supports BSR and new data items information forwarding. In this scenario it is recommended to never set “Send Codeset 6/7 LAI IE” to “y” in order to save SETUP message space and to ensure operation over networks that do not allow codeset 6/7 IEs. This combination is the recommended setup for Multi-Site Routing.

Appendix G: Functional differences for DEFINITY G2 and DEFINITY ECS Call Vectoring and EAS

Introduction

This appendix provides the Call Vectoring functional differences between the DEFINITY Generic 2 (G2) switch and the Avaya MultiVantage system. This information should prove helpful to system administrators who administer networks that use both the DEFINITY G2 and MultiVantage.

The appendix is organized into three sections:

- Differences in command function on page 551
- General Call Vectoring Functional Differences on page 557
- Differences in defining/interpreting split flows on page 560
- EAS differences on page 561

Note:
Call Prompting is not supported on DEFINITY G2.

Differences in command function

The following sections indicate the differences for Call Vectoring commands between the two systems. The commands discussed include the following:

- **`queue-to split`**
- **`check split`**
- **`goto step`**
- **`goto vector`**
- **`route-to number`**
- **`announcement`**
- **`wait-time`**
- **`busy`**

queue-to split and check split

The **`queue-to split`** command queues the call to the specified split and assigns a queuing priority level.

The **`check split`** command checks the status of a split for possible termination of the call to that split. When termination is not possible, queuing at the specified priority is attempted. Termination and/or queuing is attempted if the split meets certain conditions that are specified as part of the command.

Differences for queuing commands

MultiVantage	DEFINITY G2
The call is simultaneously queued to a maximum of three different splits. The indicated split is checked only once, and if the specified condition is met, an attempt to terminate or queue the call is made. Multiple checking of a backup split requires repeating the **`check split`** command multiple times and/or unconditional **`goto step`** looping. After the call is queued to three splits, subsequent queue commands in the vector for additional splits fail and are skipped (unless these commands specify a different priority).	The call is queued to one split at a time. Successful queue commands that occur after the call is already queued cause the call to be dequeued from the first split and queued to a new split. Each **`check`** step executed by vector processing is rechecked in the background every two seconds while the steps that follow are processed. This process continues until the specified conditions are met. The periodic threshold checking of the **`check split`** commands is implemented to simulate multiple split queuing.
Calls can be queued to vector-controlled splits via Call Vectoring or to ACD splits/hunt groups directly via hunt group/split extensions when vectoring/prompting is active. Vector-controlled splits can be directly accessed via split extensions or via **`route-to`** commands to the extension ACD splits/hunt groups can also be accessed via **`route-to`** commands to the extension.	When Call Vectoring is active, calls can be queued to ACD splits only via the **`queue to main split`** and **`check split`** Call Vectoring commands.
Calls cannot be queued to splits that already hold the number of queued calls defined by the split queue size on the hunt group screen. Therefore, every queuing command should be preceded by a check step to determine if the queue is full. Also, queue limits should be set as high as possible to ensure the call queues.	No split queuing capacity limits are in effect, and the commands are never skipped.
The **`check split`** command can test a maximum threshold of 999.	The command can test a maximum threshold of 99 calls.

Differences for queuing commands (continued)

MultiVantage	DEFINITY G2
The oldest-call-waiting test condition within the **`check split`** command has a range of 1 through 999 seconds in one-second increments. An unconditional **`check split`** command is allowed.	The oldest-call-waiting test condition within the **`check split`** command has a range of 0 through 999 seconds in one-second increments. The **`check split`** command is conditional only.
The **`rolling-asa`**, **`expected-wait`**, and **`wait-improved`** conditions are available with the **`check split`** command.	These capabilities are not provided.
The **`queue-to`** and **`check`** commands can queue a call to the best resource as determined by a series of **`consider`** steps.	These capabilities are not provided.

goto step and goto vector

The **`goto step`** command allows conditional or unconditional movement (branching) to a preceding or subsequent step in the vector.

The **`goto vector`** command allows conditional or unconditional movement (branching) to another vector.

Differences for goto commands

MultiVantage	DEFINITY G2
The commands can test a maximum threshold of 999.	The commands can test a maximum threshold of 99 calls.
The oldest-call-waiting test condition within the commands contains a range of 1 through 999 seconds and is checked according to a 1-second increment.	The oldest-call-waiting test condition within the commands contains a range of 0 through 999 seconds and is checked according to a 1-second increment.

Differences for goto commands

MultiVantage	DEFINITY G2
The **`rolling-asa`**, **`expected-wait`**, **`counted-calls`**, **`ani`**, **`ii-digits`**, and **`interflow-qpos`** conditions are available with the **`goto`** commands. Vector routing tables can be checked for the **`digits`**, **`ani`** and **`ii-digits`** conditions.	These capabilities are not provided.
The **`goto...if expected-wait`** commands can use the **`best`** keyword and **`wait-improved`** condition.	These capabilities are not provided.

route-to number

This command routes the call to a specific number.

Differences for route-to number command

MultiVantage	DEFINITY G2
The actual digit string is used as the destination. The string can contain special characters that may be stored in an AD string, including *~p, ~w, ~W, ~m,* and *~s* (but not * or #). (See the **`route-to number`** command in the manual pages of Appendix A: Call Vectoring commands on page 387.) Feature access codes (AAR/ARS) or trunk access codes may be used to route calls externally.	The AD member number is used as the destination. None of the special characters may be used. The special functions are handled by the AAR/ARS pattern routing.
The trunk may be accessed via ARS/AAR, TAC, or UDP.	AAR/ARS is required for non-DCS trunk calls.
Routing to individual attendant extensions is permitted.	The individual attendant extension feature is not available.
Routing to announcement extensions is permitted.	The announcement command is required for all announcement access.
If the command fails, and if the command is the last step in the vector, the command is not retried. If retrying is required, an unconditional goto step can be used to loop back to the route-to step.	If the command is the last step in the vector, a busy destination targeted by the command is retried every two seconds.

Differences for route-to number command (continued)

MultiVantage	DEFINITY G2
The command with the interflow-qpos condition tests the call for interflow eligibility.	This capability is not provided.
Routing to an ACD split extension is allowed even if Call Vectoring is operational.	This capability is not provided.
Routing to a Service Observing FAC is allowed	This capability is not provided.

announcement

This command indicates that the caller should expect to hear an announcement. Although the DEFINITY G2 announcement strategy differs from the MultiVantage announcement strategy, each one assures that, theoretically, the entire announcement is played from the beginning.

Differences for announcement command

MultiVantage	DEFINITY G2
Announcement extensions are used.	Announcement numbers are used.
Provides integrated board internal announcements.	Integrated announcement board is not supported.
The system supports auxiliary trunk-connected external announcement devices.	Supports only auxiliary trunk-connected announcement devices.
The maximum number of calls that can be queued and connected to an announcement is limited by preassigned queue slots. The system allows for multiple integrated announcement boards. See *Administrator Guide for Avaya MultiVantage Software*, 555-233-506 for details.	Limited only by the number of time slots available on the module to which the announcement channel is connected. The maximum number of time slots is 256.

wait-time

This command sets a length of time for a call to wait in the queue. The command also specifies one of the following treatments while the call advances in the queue(s):

- Silence
- Ringing
- Music
- I-silent
- Alternate Audio/Music Source)

Differences for wait command

MultiVantage	DEFINITY G2
The system-wide music-on-hold feature must be active for music treatment on the command. An alternate audio/music source can be administered for a **`wait-time`** step.	A separate music option is available for Call Vectoring.

busy

This command terminates vector processing and gives the caller a busy signal.

Differences for busy command

MultiVantage	DEFINITY G2
A timeout after 45 seconds is provided.	A 20 second timeout is provided for both CO and non-CO trunks.

General Call Vectoring Functional Differences

The following table provides an overview of general differences for Call Vectoring operations between the DEFINITY G2 and MultiVantage.

General call vectoring functional differences

Topic	MultiVantage	DEFINITY G2
General ACD	Split queue size is administered on a per split basis with a system-wide maximum of calls. Call queue space for the appropriate maximum number of calls must be distributed on a preassigned basis over all assigned hunt groups and (vector-controlled or nonvector-controlled) ACD splits.	There is no limit to the size of individual split queues.
	An agent may be concurrently logged into three splits at a time.	An agent may be logged into only one split at a time.
	The agent hears the same zip tone signal for calls that are queued to the main split as well as for intraflowed/interflowed calls.	One burst zip tone is provided for calls that are queued to the main split. Two burst zip tones are provided for intraflowed calls (via the **`check split`** command), and three burst zip tones are provided for interflowed calls (via Look-Ahead Interflow).
ACD Split Strategy	A split or a hunt group can be accessed by either a call vector or a group extension. This allows for both vector calls and nonvector calls in a single split's queue.	When Call Vectoring is optioned, splits do not have extensions. All access to splits must go through a Call Vector via **`queue to main split`** or **`check split`** commands.
	Non-vector-controlled splits can specify redirection treatment (such as Call Coverage, Call Forwarding, etc.) and announcement treatment.	Only vector-controlled splits are available when Call Vectoring is active.

General call vectoring functional differences (continued)

Topic	MultiVantage	DEFINITY G2
VDN Access/ Capacity	COR checking is used for access to a VDN and for routing to a station.	No restriction checking is used to access a VDN. **NOTE:** Both systems use the Facility Restriction Level (FRL) associated with the VDN for outgoing trunk calls.
	COR checking is used when routing locally from a vector.	No restriction check is implemented for local routing.
	A maximum of 20000 VDNs can be used.	The maximum number of VDNs is limited only by the number of extensions capacity (32K).
Voice Mailbox	**`messaging split`** command is used.	Calls are routed to a messaging split via a route to another VDN assigned to a vector with a queue to AUDIX.
Miscellaneous	Changes made to vector administration take effect upon submission. These changes can affect current calls.	A “scratch” pad is used for vector changes. Consequently, only new calls that enter the vector receive the treatment specified in the corrected vector. Vector processing for existing calls is completed in the old vector.
Miscellaneous (continued)	An existing vector can not be copied to another blank vector. (This capability, is available via CMS administration.)	These capabilities are provided by the switch administration.
	Either the VDN or the final destination (but not both) is provided in the CDR record.	Variable format CDR (formerly SMDR) records can be used. Consequently, both the VDN and the final destination can be provided. **NOTE:** CDR records allow the VDN to be specified in the calling party field.
	Blank steps are allowed in vectors, and blank vectors (with no steps defined) may exist.	Blank steps or blank vectors are not allowed (CMS also does not support this).

General call vectoring functional differences (continued)

Topic	MultiVantage	DEFINITY G2
	Trunk groups can be assigned to VDNs only via switch administration.	Trunks groups can be assigned to VDNs via CMS administration.
	Vector processing is limited to a maximum of 1000 step executions for a call (limit increased to 3000 with **`interflow-qpos`** in vector). Once this maximum is reached, processing stops. There is an implied wait of 0.2 seconds for every seven executed steps.	Separate 1000 step counters are provided for execution of **`goto step`** commands and **`check split`** retries. If either counter exceeds 1000, the call is forced disconnected. Only **`check split`** retries are counted on internal calls.

Differences in defining/interpreting split flows

Split flows are defined and/or interpreted according to the switch version and the management system involved. The following sections illustrate how split flow interpretation differs between the two systems as interpreted by CMS.

Note:

BCMS is not available on the DEFINITY G2 (with or without vectoring).

CMS standards for interpreting split flows

Flow type	MultiVantage with vectoring	DEFINITY G2 with traditional ACD
Inflow	Calls answered by a split other than a primary split. **NOTE:** A primary split is the first split to which a call queues.	Calls that intraflow from one split's queue to another split's queue (that is, calls that queue to a split after having been previously queued to another split).
Outflow	Calls that are dequeued from a primary split via a **`route to`** or **`messaging split`** command, or by being answered by an agent in another split to which the call is also queued.	Calls that are taken out of a split's queue and then sent to another destination.
Dequeue	Calls that are dequeued from any split other than the primary split in a VDN.	Not used.

When a call is not answered (due to an outflow, abandon, busy, or disconnect), the call's disposition is tracked for the primary split. On CMS, the other splits to which the call is queued tracks a dequeue when the call outflows, abandons, is given busy treatment, or is disconnected.

If the primary split in a VDN is unmeasured, a(n) outflow, abandon, busy, or disconnect is not tracked for the call. Also, an answer is not tracked if the call is answered by an agent in the primary split.

EAS differences

This section lists the differences between the systems for EAS.

- The DEFINITY G2.2 does not have logical agent capabilities.
  - Agent extensions are preassigned to default skill groups (groups ending in zero).
  - Agents sharing a telephone must have the same default skill group.
  - The station extension is used to provide a name, COR, and coverage path.
- MultiVantage logical agent provides the following:
  - Any station can be used as an ACD terminal for any skills.
  - Agents can be reached by dialing their login IDs.
  - Name, COR, and coverage path follow the agent to the telephone to which they are currently logged in.
- The DEFINITY G2.2 does not support Direct Agent Calling.
- The DEFINITY G2.2 does not support Call Prompting.
- The DEFINITY G2.2 login procedure is: dial feature access code, dial login ID twice. The MultiVantage login procedure is: dial feature access code, dial login ID, dial optional password.
- The DEFINITY G2.2 restricts agents with multiple skills to skills in the same skill tens group (for example, skill 20-29). MultiVantage allows an agent to be in any combination of skills.
- The DEFINITY G2.2 restricts calls queuing to multiple skills simultaneously to skills in the same skill tens group. This also applies to VDN skills. MultiVantage allows calls to queue to any three skills simultaneously.
- The DEFINITY G2.2 administers agents to a default skill and the agents enter their other skills after logging in. MultiVantage administers all of the agents' skills, and the agents are logged into all of their assigned skills during login. MultiVantage agents can change their skills.
- CMS can only change an agent's default skill on the DEFINITY G2.2 (when the agent is unstaffed). CMS can change all skills for an agent on MultiVantage (change affected the next time the agent logs in).
- The DEFINITY G2.2 does not support skill levels for agents. This also implies that the DEFINITY G2.2 does not support expert agent distribution (EAD). MultiVantage does support skill levels for agents and EAD.
- On the DEFINITY G2.2, when a change is made to a VDN skill preference, only new calls to the VDN will be impacted by the change. On MultiVantage, when a change is made to a VDN preference, existing calls will be impacted as they encounter a vector step that references the VDN skill preference.

Appendix H: Call Vectoring/EAS and BCMS/CMS interactions

Call Vectoring and Expert Agent Selection (EAS) interact with a management information system that helps to monitor and report on the activity within Call Vectoring and EAS. In most cases, the management system is either the Call Management System (CMS) or the Basic Call Management System (BCMS).

The CMS, which resides on an adjunct processor, collects and processes ACD information to generate reports. BCMS, which resides on the switch, also collects ACD information and generates a limited number of reports. The CMS reporting and data storage capabilities are much more extensive than those of the BCMS.

BCMS collects and processes ACD information to generate various reports.

This appendix is intended to illustrate how this system interprets these management systems interpret and reports report on activity within Call Vectoring and EAS. Special emphasis is placed on interpreting and reporting on this activity as it occurs within splits during a series of Call Vectoring or EAS events.

Note:

Appendix A: Call Vectoring commands on page 387 provides a summary of the CMS/BCMS interactions with each Call Vectoring command (where applicable).

CMS/BCMS tracking in a Call Vectoring environment

Tracking is the identifying of call flows and other actions relevant to call handling. There are three classes of call flows: split flows, VDN flows, and vector flows. We are most concerned with tracking in the Call Vectoring environment. The specific types of call flows and actions in this environment that are tracked by the CMS/BCMS include the following:

- Inflows (flow ins)
- Outflows (flow outs)
- Dequeues
- Abandons
- Answers
- Busies
- Disconnects

The split supervisor can use VDN and vector flows to evaluate how effective vector programming is at the site in question. The supervisor can use split flows to determine the manner in which the splits at the site are handling incoming telephone calls.

Defining and interpreting call flows

The manner in which specific call flows are defined and interpreted depends upon the call flow class in question, the management system in effect, and the version of the switch being used. Management systems include CMS and BCMS.

The following sections define and interpret specific call flows according to these parameters.

Answered and abandons

The most important tracking items for most VDNs and vectors are the number of calls answered and the number of calls abandoned. The CMS provides VDN profiles that show when calls are answered and abandoned. Ten service level intervals are administered for these profiles. These intervals can have smaller time intervals around the time most calls are answered and when most calls abandon to get more detailed information.

This data can be used to determine what an acceptable service level is for most callers. The percentage answered within the administered acceptable service level is also shown on the Call Profile reports. For VDNs, the calculation is ACD calls answered and nonACD calls connected within the service level divided by calls offered to the VDN (including calls that inflow to the VDN).

For split/skill statistics, the calculation is ACD calls answered within the service level divided by calls queued to the split/skill (answered calls, abandoned calls, calls that flow out, calls that dequeue). In most cases the VDN percentage will be higher then the split percentage since calls dequeued from a split/skill are counted as answered, abandoned, or outflows for the VDN.

Changes made to a vector or to staffing will typically affect the VDN call profile. Even the wording of an announcement can affect the abandon profile. It is worthwhile to review the VDN's call profile before and after any change to determine if the change had a positive impact.

Busies and disconnects

Busy calls and forced disconnects reported on the CMS indicate how many calls this VDN/vector turned away. If forced disconnect is used out of business hours, this item would indicate how many customers expected you to be operating during a specific time interval. If busies are given when the queues are full or waiting times are long, the number of busies in an interval might suggest a staffing change is needed. If disconnect is used to deny a look-ahead interflow attempt, a large number of denials would indicate a busy time at multiple sites.

VDN inflows and outflows

The following section discusses the specific VDN flows for CMS and BCMS.

CMS and BCMS standards

The following table illustrates how CMS and BCMS interprets specific VDN flows from the switch:

CMS and BCMS standards for interpreting VDN flows

Flow type	Management system	Interpretation
VDN flow in	CMS	Calls that flow into the VDN via a **`route-to VDN`** command or by Redirection on No Answer to a VDN.
	BCMS	(Not tracked.)
VDN flow out	CMS	Calls that successfully flow out of a VDN to another VDN or to an external location via a **`route-to`** command.
	BCMS	Same as for CMS.

Vector inflows and outflows

The following section discusses the specific vector flows as recorded by the CMS.

CMS standards

Vector flow in pertains to calls that flow into a vector from another vector via a route to or a goto vector command. Vector flow out pertains to calls that successfully flow out of a vector via a route to or a goto vector command.

Split inflows, outflows, and dequeues

The following sections discuss the various split flow types for CMS and the BCMS.

CMS and BCMS standards

The CMS and the BCMS are grouped together because both of these systems interpret two split flow types identically. These flows include inflow and outflow. The CMS interprets another split flow type, dequeue. The BCMS does not interpret this split flow type because it does not have a dequeue tracking item. This means that in a situation where the CMS tracks a dequeue, BCMS does not because it is unable to do so.

Before we detail how the CMS and the BCMS interpret split flows, we should discuss the term primary split, since this concept plays a significant role in tracking. Primary split is defined as the first split in a VDN to which a call actually queues. Therefore, this split is not necessarily the first split referenced in the vector.

Another split becomes the primary split if either of the following events occurs:

- Call cannot queue to the originally-targeted split because the split has no queue slots available.
- Call leaves the VDN (via a **`route-to`** VDN command, for example) and is queued to another split as a result.

If the call leaves vector processing and does not queue to another split (as a result of a **`route-to extension`** command, for example), there is no new primary split.

With this discussion in mind, let’s take a look at the following table to see how CMS and BCMS interpret split flows for the switch:

CMS and BCMS standards for interpreting split flows

Flow type	Management system	Interpretation
Inflow	CMS	Calls that ring at an agent in a split other than the primary.
	BCMS	Same as for CMS.

CMS and BCMS standards for interpreting split flows (continued)

Flow type	Management system	Interpretation
Outflow	CMS	Calls that are dequeued from a primary split via a **`route-to`** or **`messaging split`** command, or by ringing at or being answered by an agent in another split to which the call is also queued.
	BCMS	Same as for CMS.
Dequeue	CMS	Calls that are dequeued from and not answered by any split other than the primary split in a VDN.
	BCMS	Not tracked.

When a call is not answered (due to an outflow, abandon, busy, or disconnect), the call's disposition is tracked for the primary split as long as the call is still queued when the call abandons, outflows, etc. However, if the call abandons or outflows from ringing, the disposition is recorded for the split for which it was ringing. On the CMS, the other splits to which the call is queued track a dequeue when the call outflows, abandons, is given busy treatment, or is disconnected.

If the primary split in a VDN is unmeasured, an outflow, abandon, busy, or disconnect is not tracked for the call. Also, an answer is not tracked if the call is answered by an agent in the primary split.

Examples of split flow tracking

The following sections provide some examples of tracking in CMS and BCMS. Each section first presents a scenario of Call Vectoring events. The scenario is then followed by a table in which the tracking for the various splits involved is recorded. Following each "tracking table," an explanation of the tracking procedure is provided.

The scenarios presented include the following:

- Call answered by a primary split.
- Call answered by a nonprimary split.
- Call abandoned from queue.
- Call answered by a primary split after a route to VDN.
- Call answered by a nonprimary split after a route to VDN.
- Call answered after a route to split.

Note:

Inflows, outflows, and dequeues are not tracked for splits administered by the **`converse-on split`** command. However, if a call is answered both by a converse split and (subsequently) by a nonconverse split, an “answer” is tracked for each split. However, a call is really considered “answered” only when it is answered by a nonconverse split. Therefore, traffic measurements for converse splits should be used only to measure converse split traffic and not to calculate the total number of calls.

Call answered by a primary split – The following scenario involves a call answered by the primary split. The scenario is as follows:

1. Call comes into a VDN whose vector queues the call to splits 1, 2 and 3.
2. Call is answered in split 1.

The following table shows the tracking table for this scenario:

Tracking for call answered by primary split

	Split tracking		
	1	**2**	**3**
CMS	answer	dequeue	dequeue
BCMS	answer		

Comments:

- **CMS**: Dequeue is tracked in split 2 as well as in split 3 because the call is answered by the primary split (split 1) and is thus dequeued from splits 2 and 3 without being answered in these splits.
- **BCMS**: No dequeue tracking item is available.

Call Answered by a non-primary split – The following scenario involves a call answered by a nonprimary split. The scenario is as follows:

1. Call comes into a VDN whose vector queues the call to splits 1, 2 and 3.
2. Call is answered in split 2.

The following table shows the tracking table for this scenario:

Tracking for call answered by non-primary split

	Split tracking		
	1	**2**	**3**
CMS	outflow	inflow answer	dequeue
BCMS	outflow	inflow answer	

Comments:

- **CMS**: Outflow is tracked in split 1 because the call is answered by an agent in another split to which the call is queued (that is, split 2). Although the call is obviously removed from split 1 after it is answered in split 2, dequeue is not tracked in split 1 because split 1 is the primary split. Inflow is tracked in split 2 because the call is answered in this split and the split is not the primary split. Dequeue is tracked in split 3 because the call is removed from the split without being answered there. When the call is removed from split 3, outflow is not tracked in split 3 because this split is not the primary split.
- **BCMS**: Outflow is tracked in split 1 because the call is answered by an agent in another split to which the call is queued (that is, split 2). Inflow is tracked in split 2 because the call is answered in this split and the split is not the primary split. When the call is removed from split 3, outflow is not tracked in split 3 because this split is not the primary split.

Call Abandoned – The following scenario involves a call abandoned by the caller. The scenario is as follows:

1. Call comes into a VDN whose vector queues the call to splits 1, 2, 2 and 3.
2. Call is abandoned.

The following table shows the tracking table for this scenario:

Tracking for Abandoned Calls

	Split Tracking		
	1	**2**	**3**
CMS	abandon	dequeue	dequeue
BCMS	abandon		

Comments:

- **CMS**: Abandon is tracked in split 1 because this split is the primary split. Dequeue is tracked in splits 2 and 3 because the call is dequeued from these splits without being answered in either split.
- **BCMS**: Abandon is tracked in split 1 because this split is the primary split. Tracking is not recorded in splits 2 and 3 because no dequeue tracking item is available.

Call answered by a primary split after a route to VDN – The following scenario involves a call answered by the primary split after a **`route-to`** VDN command is executed. The scenario is as follows:

1. Call comes into a VDN whose vector queues the call to splits 1, 2 and 3.
2. Vector executes a **`route-to VDN`** step.
3. Call is then queued to splits 4, 5 and 6.
4. Call is answered in split 4.

The following table shows the tracking table for this scenario:

Tracking for call answered by primary split after route to VDN

	Split tracking					
	1	**2**	**3**	**4**	**5**	**6**
CMS	outflow	dequeue	dequeue	answer	dequeue	dequeue
BCMS	outflow			answer		

Comments:

Split 1 is the original primary split, because this is the first split to which the call actually queues. However, split 4 becomes the new primary split because:

- Call leaves the original VDN upon execution of the **`route-to VDN`** step.
- Split 4 is the first split to which the call queues upon execution of this step.
- **CMS**: Outflow is tracked in split 1 because this split is the original primary split, and the call is dequeued from this split via a **`route-to VDN`** step. Dequeue is tracked in splits 2, 3, 5, and 6 because the call is dequeued from each of these splits without being answered in any one of them.
- **BCMS**: Outflow is tracked in split 1 because this split is the original primary split.

Call answered by the non-primary split after a route to VDN – The following scenario involves a call answered by the nonprimary split after a **`route-to VDN`** command is executed. The scenario is as follows:

1. Call comes into a VDN whose vector queues the call to splits 1, 2 and 3.
2. Vector executes a **`route-to VDN`** step.
3. Call is then queued to splits 4, 5 and 6.
4. Call is answered in split 5.

The following table shows the tracking table for this scenario:

Tracking for call answered by non-primary split after route to VDN

	Split tracking					
	1	**2**	**3**	**4**	**5**	**6**
CMS	outflow	dequeue	dequeue	outflow	inflow answer	dequeue
BCMS	outflow			outflow	inflow answer	

Comments:

- **CMS**: Outflow is tracked in split 1 because this split is the original primary split, and the call is dequeued from this split via a **`route-to`** VDN step. Dequeue is tracked in splits 2, 3, and 6 because the call is dequeued from each of these splits without being answered in any one of them. Outflow is tracked in split 4 because this split becomes the new primary split after the **`route-to VDN`** step is executed and the call is subsequently dequeued from this split by being answered in another split (split 5) to which the call is also queued. Finally, inflow is tracked in split 5 because the call is answered in this split, and the split is not the primary split.
- **BCMS**: Outflow is tracked in split 1 because this split is the original primary split. Outflow is tracked in split 4 because this split becomes the new primary split after the **`route-to VDN`** step is executed. Finally, inflow is tracked in split 5 because the call is answered in this split, and the split is not the primary split.

Call answered after a route to split – The following scenario involves a call answered after it is routed to a split via a **`route-to digits`** or **`messaging split`** command. The scenario is as follows:

1. Call comes into a VDN whose vector queues the call to splits 1, 2 and 3.
2. Vector executes a **`route-to digits`** (or **`messaging split`**) step.
3. Call is queued to split 4 and answered by an agent in split 4.

The following table shows the tracking table for this scenario:

Tracking for call answered after route to split

	Split tracking			
	1	**2**	**3**	**4**
CMS	outflow	dequeue	dequeue	answer
BCMS	outflow			answer

Comments:

- **CMS**: Outflow is tracked in split 1 because this split is the original primary split, the call is dequeued from this split via a **`route-to digits`** (or **`messaging split`**) step, and the call is answered in split 4, which becomes the new primary split. Dequeue is tracked in splits 2 and 3 because the call is dequeued from each of these splits without being answered in any one of them.
- **BCMS**: Outflow is tracked in split 1 because this split is the original primary split, and the call is answered in split 4, which becomes the new primary split.

Evaluating split performance

By using the information presented to this point, along with the information from various reports (as discussed in the next section), the split supervisor can answer one or more questions concerning split performance and then make adjustments, if necessary. Here are some of the questions the supervisor can answer:

1. How many ACD calls offered to my split were “mine” (that is, were offered to this split as the primary split)?

   Note:

   Split “ACD calls” include Direct Agent Calls for BCMS, but not for CMS, which tracks Direct Agent Calls separately.

2. How many of “my” ACD calls did “my” split not answer?
3. How many ACD calls that I didn’t answer weren’t “mine?”

The following sections present the answers to these questions from the perspective of the CMS and BCMS.

CMS – The following answers reflect the use of the CMS:

- The number of calls offered to “my” (primary) split that were “mine” can be determined via examination of the CMS Split Summary Report. The algorithm is as follows: CALLSOFFERRED - INFLOWCALLS - DEQUECALLS (that is, the total number of calls offered minus the number of calls not “mine” that I answered minus the number of calls not “mine” that I didn’t answer.)
- The number of “my” calls that “my” split didn’t answer can be determined via examination of the CMS VDN Report. The algorithm is as follows: ABNCALLS + BUSYCALLS + DISCCALLS + OUTFLOWCALLS (that is, the number of abandoned calls plus the number of busy calls plus the number of disconnected calls plus the number of calls outflowed from “my” split tagged as a primary split).
- The number of calls not “mine” that “my” split didn’t answer is DEQUECALLS, which is indicated in the CMS Split Summary Report.

BCMS – The following answers reflect the use of BCMS:

- The number of calls offered to “my” split that were “mine” can be determined via examination of the BCMS Split Report. The algorithm is as follows: ACDCALLS + ABNCALLS + OUTFLOWCALLS - INFLOWCALLS (that is, the total number of calls answered plus the total number of calls abandoned from “my” split tagged as a primary split plus the number of calls that outflowed “my” split tagged as a primary split minus the number of calls answered that were not directed to “my” split tagged as a primary split).

Using CMS and BCMS reports to evaluate Call Vectoring activity

There are a number of CMS and BCMS reports that allow you to evaluate Call Vectoring activity. Some of these facets include the call flows present within Call Vectoring as well as the speeds at which calls are answered. The sections that follow identify and discuss the CMS and BCMS reports that indicate this activity.

CMS reports

CMS has real-time, historical, and integrated reports. Most of the CMS historical reports are available in four versions: intra-hour, daily, weekday, and monthly. The following list identifies and describes several CMS reports that summarize Call Vectoring activity. For further details on these and other related reports, see *Avaya Call Management System Supervisor Version 11 Reports*, 585-210-708.

Note:

The reports described in this section are generated in CMS R3 and newer releases of the CMS. Corresponding CMS R2 reports do not provide information that reflects capabilities that are new to the switch (for example, internal/external call tracking).

- **Split Summary Report** summarizes the call activity for an entire split. Among other information, the report provides the number of calls answered, the total number of flow ins (inflows), flow outs (outflows), dequeues, and abandoned calls.

  The report also indicates the average speed of answer (interval ASA) for calls. This refers to the sum of the queue time and ring time for a call within the answering split only. Finally, the report indicates the dequeued average queue time, which is the average time a call waits until it is answered by another split to which the call is also queued.

- **VDN Report** summarizes VDN activity for specific vectors. Among other information, the report provides calls answered, connected, abandoned, the number of VDN Flow Ins/Outs, calls forced busy, and calls forced disconnect. VDN Flow In pertains to calls that flow into a VDN from another VDN via a **`route-to`** command. VDN Flow Out pertains to calls that successfully flow out of VDN to another VDN or external location via a **`route-to`** command.
- **Vector Report** summarizes vector activities. Among other information, the report provides the number of calls offered, calls answered, calls abandoned, Vector Flow Ins/Outs, calls forced busy, and calls forced disconnect. Vector Flow In pertains to calls that flow into a vector from another vector via a **`route-to`** or **`goto vector`** command. Vector Flow Out pertains to calls that successfully flow out of a vector via a **`route-to`** or **`goto vector`** command.

BCMS reports

BCMS has a real-time split report, split historical reports, real-time VDN reports, and VDN historical reports. The following list identifies and describes several BCMS reports that summarize Call Vectoring activity. For more information on these and other related reports, refer to *Avaya MultiVantage Basic Call Management System (BCMS) Operations*, 555-230-706.

BCMS Split Report – Summarizes the call activity for an entire split. The information can be requested either daily or by the administered time period. Among other information, the report provides the total number of flow ins (inflows) and flow outs (outflows), the calls answered and calls abandoned. The report also provides the average speed of answer time for calls handled by the split during the indicated time period.

VDN Summary Report – Summarizes statistical information for all internally-measured VDNs. The information can be requested by the administered time interval or daily. The **`list bcms vdn`** report gives multiple time periods or days for a single VDN. The **`list bcms summary vdn`** report gives a one-line summary per vdn (with data from the specified times or days), but can give the data for numerous vdns.

The report also indicates the total number of flow outs, specifically, the number of calls that route to another VDN or to a destination external to the switch. However, calls that encounter a **`goto vector`** command are not shown as outflows. No further measurements are taken on the calls once the calls have outflowed. If an outflowed call later abandons, this is not indicated in the report.

Among other information, the VDN report provides a total for offered calls, answered calls, abandoned calls, and also one for calls that were either **`forced busy`** or **`forced disconnect`**.

VDN Real-Time Report – Provides statistical information including the number of calls currently waiting and the oldest call waiting. The VDN real-time report has the same characteristics as other real-time BCMS reports.

Using CMS in an EAS environment

The same tracking and database items used within a traditional Call Vectoring environment are used within an EAS environment but there are also new items that are specific to EAS. All existing custom reports should work when you are upgrading to EAS.

Tracking entities

The following sections explain how the following entities are tracked in an environment with EAS optioned:

- Agents and their skills
- Direct Agent Calls
- Non-ACD Calls
- VDN Skill Preferences

Agents and their skills

The fields under the “Extn” column in the CMS Real-Time Agent Report show the extension that the agent is logged into. These fields can be used to locate the agent or to service observe the agent.

With EAS optioned, the Skill Status Report replaces the Split Status Report. This report indicates the skills logged into and the skill level of each skill. If too many calls are waiting, or if calls are waiting too long (also shown on the Skill Status report), it is possible that not enough agents have the skill administered at a high enough skill level.

An agent may be denied login to some skills if the maximum agents/skill number is met or if the CMS limit on agent/skill pairs logged in has been reached.

The Login/Logout Historical Report also lists up to 15 of the agent’s skills and the skill levels for each.

Direct Agent calls

Waiting Direct Agent calls are not included in the “Calls waiting” and “Oldest Call Waiting” report fields for skills because such calls are not skill calls. However, Direct Agent Calls are included in these two report fields for VDNs.

The Queue/Agent Summary Real-Time Report lists separately the Direct Agent calls waiting in a skill queue. Direct Agent calls are queued to the skill that is administered as the Direct Agent Skill. To manage the skill’s queue slots effectively, it is recommended that a skill be dedicated for Direct Agent calls.

Since Direct Agent calls are not skill calls, the skill tables do not track Direct Agent calls; however, the tables do monitor skill queue slots. The agent's time is tracked as “OTHER” in the skill tables. In the agent tables, there are separate Direct Agent call items. The standard CMS agent reports add the Direct Agent calls and the skill ACD calls and report these calls as “ACD Calls.” The VDN tables track Direct Agent calls as ACD calls.

Non-ACD calls

The first measured skill that an EAS agent is logged into is used by CMS to track non-ACD calls unless the agent has an ACD call on hold. If an ACD call is on hold, outgoing non-ACD calls are counted for the skill of the held ACD call.

VDN skill preferences

VDN skill preference data is collected to provide information on what groups of agents (skills) are handling calls and on how effectively each skill group handles a particular VDN.

Real-time and historical VDN Skill Preference reports can be used to compare the percentage of calls being answered by the 1st, 2nd, and 3rd VDN preferences against an objective. If too few calls are being answered by the 1st skill preference, the vector can be adjusted to allow more time for the 1st skill preference group to answer calls; another alternative is to train or hire more agents with the 1st skill preference.

You can use VDN skill preference data to compare the average talk time and average ACW time for agents in the 1st, 2nd, and 3rd skill groups. If these times vary too much across groups, more training may be needed for the backup groups (that is, the 2nd and 3rd skill groups).

VDN skill preference data is tracked according to the skill preferences (1st, 2nd, 3rd) assigned to the VDN. Whenever a vector step either references a 1st, 2nd, or 3rd skill or specifies a skill number that matches the 1st, 2nd, or 3rd skill administered, the new database items are tracked. For example, if VDN 1000 has Skills 21, 22, and 23 administered as the 1st, 2nd, and 3rd skills, respectively, and if the vector associated with VDN 1000 has a “queue to main skill 22" step, tracking occurs for the 2nd VDN skill preference if the call is answered by an agent in Skill 22. Skill preference tracking also occurs for Skills 21 and 23. This allows users who prefer to specify the actual skill number in the vector to take advantage of the tracking for VDN skill preferences.

EAS administration from CMS

CMS can be used to administer vectors as well as skills for agents and VDNs. The ACD Administration: Change Agent Skills CMS screen is used to display and modify the skills and levels assigned to an agent, as well as the assigned Direct Agent skill and call handling preference.

The ACD Administration: Change VDN Skill Preferences screen is used to request a VDN's skill preferences and to modify the VDN's skills.

The CMS Vector Contents screen is used to create and modify vectors. CMS supports the Call Vectoring commands that queue calls to the 1st, 2nd, or 3rd VDN skill.

Appendix I: Operation details for the route-to command

The **`route-to`** command can be programmed with or without coverage. The following table summarizes the operation of the **`route-to`** command for each of the destination types and conditions associated with the commands.

Switch route-to command operation

Condition	cov = n Any Step	cov = y Any Step[1]
Invalid Destination[2]	Goes to next step, else stop	Goes to next step, else stop
VDN Extension[3]		
Vector Assigned	Goes to new vector	Goes to new vector
Vector Has No Steps	Stop[4]	Stop[4]
Station Extension Idle (all appearances idle)		
CF-ALL Active or CF-DA Applies	Forwards if possible, else next step, else stop[4]	Forwards if possible, else coverage, else busy
Coverage		
• DA Applies	Rings idle app.	Coverage on DA
• All Applies	Goes to next step, else stop[4]	Coverage
• SAC Applies	Rings idle appearance	Coverage
• None of Above Applies	Rings idle appearance	Call delivered and is allowed to cover

Switch route-to command operation (continued)

Condition	cov = n Any Step	cov = y Any Step[1]
Station Extension Active (with idle 2-way app)		
CF-ALL Active	Forwards if possible, else next step, else stop[4]	Forwards if possible, else coverage, else busy
Coverage		
• DA Applies	Rings idle app (no DA timing)	Coverage on DA
• Ext Act Applies	Rings idle appearance	Coverage
• All Applies	Goes to next step, else stop[4]	Coverage
• SAC Applies	Rings idle appearance	Coverage
• None of Above Applies	Rings idle appearance	Call delivered and is allowed to cover
Station Extension Busy (no idle 2-way app)		
Extension in Hunt Grp (also see ACD Hunt Grp)	Queues if possible, else next step, else stop[4]	Queues if possible, else coverage, else busy
CF-ALL Active or CF-DA Applies	Forwards if possible, else next step, else stop[4]	Forwards if possible, else coverage, else busy
Call Waiting to Analog Sta Would Apply	Goes to next step, else stop[4]	Call waits
Coverage		
• Ext Act Applies	Goes to next step, else stop[4]	Coverage
• Ext Bsy Applies	Goes to next step, else stop[4]	Coverage
• All Applies	Goes to next step, else stop[4]	Coverage
• SAC Applies	Goes to next step, else stop[4]	Coverage
• None of Above Applies (or hunt, fwd, or cov destination is unavailable)	Goes to next step, else stop[4]	Busy tone given
Extension with Incompatible COR	Goes to next step, else stop.	Goes to next step, else stop.

Switch route-to command operation (continued)

Condition	cov = n Any Step	cov = y Any Step[1]
Terminating Extension Group		
All Members Idle	Rings idle appearance	Call delivered and is allowed to cover
A Member Active on TEG	Goes to next step, else stop[4]	Coverage, else busy
No Idle App on Any Member	Goes to next step, else stop[4]	Coverage, else busy
Hunt Group Extension		
Idle Agent	Rings idle appearance	Call delivered and is allowed to cover
No Idle Agent		
• Call cannot queue	Goes to next step, else stop[4]	Busy tone given
• Call can queue	Call is queued	Call is queued
Extension on Another Node (Uniform Dialing Plan - UDP DCS or non-DCS)		
Trunk available	Call delivered	Call delivered
Trunk not available	Goes to next step, else stop[4]	Queues if possible, else reorder
No DCS Buffer for Routing	Call delivered w/o DCS msg	Call delivered w/o DCS msg
Trunk Access Code (TAC) Destination		
Trk Grp No Dial Access	Goes to next step, else stop[4]	Routes to local attendant
Trunk Available	Call delivered	Call delivered
Trunk Not Available	Goes to next step, else stop[1]	Queues if possible, else reorder

Switch route-to command operation (continued)

Condition	cov = n Any Step	cov = y Any Step[1]
AAR/ARS FAC Dest. (including Subnet Trkng)		
Trk Grp No Dial Access	Tries next route	Routes to local attendant
Trunk Available	Call delivered	Call delivered
Other Routes Avail	Call delivered	Tries next route
All Routes Busy		
• No Pattern Queuing	Goes to next step, else stop[4]	Reorder tone given
• Queuing Assigned	Goes to next step, else stop[4]	Queues to pattern
Attendant Queue (dial 0)		
Idle Atnd	Rings idle appearance	Call delivered and is allowed to cover
No Idle Atnd		
• Not In Night Svc	Call is queued	Call is queued
• In Night Svc		
— Dest. assigned	Delivered to night service	Delivered to night service
— Not assigned	Call is queued	Call is queued
Individual Attendant Access		
Attendant idle	Rings idle appearance	Call delivered and is allowed to cover
Attendant busy	Queues if possible else Goes to next step, else stop[4]	Queues if possible, else Busy tone given
CAS Attendant With Caller on Branch		
RLT available	Rings idle appearance	Call delivered and is allowed to cover
All RLTs busy	Queues if possible, else next step, else stop[4]	Queues if possible, else busy tone

Switch route-to command operation (continued)

Condition	cov = n Any Step	cov = y Any Step[1]
Inter-PBX Atnd Calling		
Trunk Grp Controlled	Routes to local atendant	Routes to local atendant
Trunk Available	Call delivered	Call delivered
Trunk Not Available	Goes to next step, else stop[4]	Reorder tone given
Look Ahead Interflow (LAI) (feature active & routes over ISDN-PRI facility)[5]		
B-Channel Not Available	Goes to next step, else stop[4]	Queues if possible, else reorder
B-Channel Available and Receiving Switch:		
• Accepts	Interflow succeeds[6]	Call cut-through
• Rejects	Goes to next step, else stop[4]	Call gets busy/disconnect
Receiving Switch w LAI Acting as Tandem Sees from Remote Receiving Switch:		
• Call Accepted	Interflow succeeds[6]	Call cut-through
• Call Rejected	Goes to next step at receiving switch, else sending switch considers call rejected after 2-minute timeout	Call gets busy/disconnect
• **`if interflow-qpos`**	Determines if queued call is eligible for interflow	Determines if queued call is eligible for interflow

1. When the **`with coverage`** option is set to **`y`**, the call is removed from vector processing when the route-to step is reached, regardless of facility or remote switch availability. The call is taken out of any split queue, and any feedback, such as music or ringback, is removed, even if the destination is not available. If the call is subsequently "rejected" by the receiving switch vector, subsequent call treatment is defined by the "rejection" command (either busy or forced disconnect). The call is treated as though the destination is directly dialed (see footnote 3 for related information). This includes coverage, forwarding, treatments for calls that cannot be completed (busy reorder, and intercept) and displays. The answering station sees only caller name and number, unless the Display VDN for route-to DACS option is enabled (for more information, see Displaying VDN names for vector-initiated Direct Agent calls on page 480). A call routed via an **`adjunct routing`** command is treated the same way as a call that is routed via a **`route-to with coverage y`** command.

2. Invalid destinations include the following: empty (for example, zero collected digits) or invalid route-to destination number, unassigned extension number, incomplete number of digits for AAR/ARS pattern, non-AAR/ARS feature access code (FAC), maintenance busy station extension, COR of the VDN that prevents access (for example, origination restricted), FRL of a VDN that is lower than required for the AAR/ARS pattern access, no routes assigned to the AAR/ARS pattern, incompatible calling and destination partitions, ACTGA trunk group destination, or an off-net forwarding destination. If a TAC (trunk access code) destination is involved, and if the TAC is for a CO/FX trunk with a **`route-to with coverage n`** step, the digits entered must match a valid ARS analysis string. If not, the destination is considered invalid. For other trunk types with a **`route-to number`** or **`route-to digits with coverage n`** step, the step succeeds when the trunk is seized (that is, vector processing stops). For a **`route-to with coverage y`** step, the step succeeds if the TAC is assigned.

3. A call that routes to a VDN via the **`route-to number with cov = y unconditionally`** command behaves like a directly- dialed call instead of a VDN call. Therefore, the terminating station's display only shows the originating station information and does not show the VDN information (for other types of VDN calls, the terminating station would see the VDN name).

4. The interaction "Stop" means the following: vector processing is stopped, the call remains queued to a split, and the caller continues to hear feedback initiated by a previous step. In the case where the **`route-to`** command fails and processing stops (due to a busy station or trunk group destination), retry can be implemented in the vector. Retrying is accomplished by including an unconditional **`goto`** step as the last step to allow for a loop back to the **`route-to`** command. Use of an intermediate **`wait-time`** command step with appropriate feedback and delay interval is strongly recommended in order to reduce processor occupancy.

5. With one exception, any **`route-to with cov= y`** step that routes over ISDN-PRI facilities cancels Look-Ahead Interflow. The exception occurs when a call reaches a vector via coverage to a VDN. Calls that cover to a VDN will not be further forwarded or otherwise redirected. For covered calls, a **`route-to`** command with coverage set to y functions as though coverage were set to n. Thus, a **`route-to with coverage y`** will route covered calls via LAI over ISDN facilities if LAI is enabled.

6. On the sending switch, the call is removed from vector processing (that is, the call is taken out of any split queue and any feedback, such as music or ringback, is removed).

Appendix J: Call flow and specifications for converse – VRI calls

This appendix details call flow for calls involving a **`converse-on`** vector step and Voice Response Integration (VRI). This call flow is segmented into the following phases:

- Converse call placement
- Data passing (optional)
- VRU data collection (optional)
- Script execution
- Data return (optional)
- Script completion
- Switch data collection (optional)

Note:

If, during any phase of this call flow, a **`converse-on`** step is executed while the caller is in the split queue and an agent becomes available to service the caller, the VRU port is dropped, vector processing is terminated, and the calling party is immediately connected to the available agent.

Converse call placement

The first action taken by the **`converse-on`** step is to deliver the call to the converse split. Ringback tone is not heard by the caller. Any audible feedback supplied by vector processing remains until the VRU answers the call and all digits (if administered) have been outpulsed to the VRU. Vector processing is suspended. Callers remain in any nonconverse split queues, and they retain their position in queue while the converse session is active.

If a Call Prompting TTR is allocated to the call, the TTR is released. Any dial-ahead digits are discarded. However, any digits collected prior to the **`converse-on`** step are kept.

Calls to busy converse splits are allowed to queue. The priority of the call in queue is administrable within the **`converse-on`** step. Again, any audible feedback supplied by vector processing continues until the call is answered by the VRU and any data is outpulsed. Calls to busy converse splits have either no queue or a full queue fail. For this scenario, a vector event is logged, and vector processing continues at the next vector step.

Whenever a **converse-on** step places a call to an auto-available split whose agents are all logged out, the call is not queued. Instead, the **converse-on** step fails, a vector event is logged, and vector processing continues at the next vector step.

Note:

Usually, this scenario occurs whenever the Voice Response Unit (VRU) goes down, the ports are members of an Auto-Available Split (AAS) and the Redirection on No Answer (RONA) feature has taken all the ports out of service.

The originator's display is not changed by the terminating or answering of a converse call. Also, whenever a call is delivered to a display station via a **converse-on** step, the station displays the following information: "Originator Name to VDN Name." Conventional Call Vectoring rules for Override are in effect.

Valid destinations for converse calls must be vector-controlled and include the following:

- Hunt groups
- ACD (including Auto-Available) splits
- Agent (including Auto-Available) skill groups
- AUDIX hunt groups

Note:

Even though AUDIX hunt groups are valid destinations for converse calls, they do not need to be vector-controlled.

Undefined and nonvector-controlled hunt group, split or skill numbers are rejected at administration time.

Any attempt to remove a hunt group, split or skill administered within a **converse-on** vector step is denied until the vector has been changed. Also, any attempt to make a hunt group, split, or skill nonvector-controlled is denied if the hunt group, split, or skill is called by a **converse-on** step.

Data passing

This phase is optional and is in effect only if the application calls for the switch to pass information in-band to the VRU.

The **`converse-on`** step may outpulse up to two groups of digits to the VRU. The digits can serve two major purposes, as follows:

- Notify the VRU of the application to be executed
- Share call-related data, such as ANI, CINFO or caller digits collected by the switch

In many applications both application selection and data sharing are required.

Since in many cases the digit strings are of variable length, the switch always appends a pound sign (#) to the end of each digit string. Prompt and collect steps in the VRU script must therefore always be administered to expect the pound sign (#) as the end-of-string symbol and to include the pound sign in the digit count.

Sending the pound sign (#) prevents excessive delays and other problems caused by digit timeouts.

The complete outpulse sequence is summarized as follows:

1. VRU answers the call
2. Delay for the time administered in the “Converse first data delay” field in the System Parameters-Features form occurs
3. <data_1> is outpulsed
4. “#” is outpulsed
5. Delay for the time administered in the `Converse second data delay` field in the System Parameters-Features form occurs
6. <data_2> is outpulsed
7. “#” is outpulsed

Note:

The length of DTMF tones (digits) and the interdigit pause between tones is administrable on the Feature-Related System Parameters form. The optimum timers for the Conversant are a 100 msec tone and 70 msec pause (administration default).

Any audible feedback supplied by the switch is disconnected only after the outpulse sequence is completed. Also, any touch-tone dialing by the calling party during the data passing phase does not result in data corruption.

The following values may be administered for <data_1> and <data_2> within the **converse-on** command:

- **Administered digit string**: This string can contain up to six characters consisting of one or more digits (0 through 9) or asterisks (*). The pound sign (#) may not be included in a digit string because it is reserved as the end-of-string character. However, a single "#" may be administered.
- **ani**: If the call is a local call or an incoming DCS call, this data type causes the extension of the calling party to be outpulsed. If the call is an incoming ISDN PRI call with ANI (BN) provided to the switch, the calling party number/billing number (CPN/BN) of the calling party is outpulsed to the voice information system. If there is no ANI (BN) to send, the end-of-string pound sign (#) is the only character outpulsed. Any other type of incoming call results in "#" being outpulsed.
- **vdn**: This data type causes the VDN extension to be outpulsed. In cases where multiple VDNs are accessed, normal VDN override rules determine which VDN extension is outpulsed.
- **digits**: This data type can be used only if Call Prompting is optioned, and it causes the most recent set of digits collected in vector processing to be outpulsed. If no digits are available, the end-of-string pound sign (#) is the only character outpulsed.
- **qpos**: This data type causes the value of the queue position of a call in a nonconverse split to be outpulsed. This value is a variable length data item from which between one and three digits can be outpulsed. If the call is not queued, the end-of-string pound sign (#) is the only character outpulsed.

  Note:

  The use of this keyword is not recommended with multiple split queuing because any queue position value sent may not be meaningful. However, if the call is queued to multiple nonconverse splits, the value of the caller's queue position in the first nonconverse split is sent.

  This data may be used by the voice information system to inform callers of their position in queue or to decide whether to execute a long or short version of a voice response script.
- **wait**: This data type sends the expected wait time for a call in vector processing that is queued to at least one split. It is a value from 0 to 9999 seconds (variable length, that is, not padded with zeros) always followed by a # digit. If the call is not queued, or is queued only to splits with no working agents, only the # is outpulsed.
- **"#"**: This is the only character outpulsed. Outpulsing this character causes the corresponding prompt and collect command in the voice response script to be skipped.
- **"none"**: This data type causes no characters to be outpulsed. Also, no end-of-string pound character (#) is outpulsed, and no time delays are invoked.

The switch always outpulses a pound character (#) at the end of each digit string. Where "#" is administered, or where the "digits" keyword is administered and the last digit collected from the caller is "#," only one "#" is outpulsed. No "#" is outpulsed when the keyword "none" is administered.

If <data_1> is administered as "none," <data_2> must also be "none."

Any data to be passed to the VRU from the switch is outpulsed in-band. Two time delays on the System Parameter-Features form (`Converse first data delay` and `Converse second data delay`) are administrable by customers. These delays may range from 0 through 9 seconds, with a default of zero seconds for the converse first data delay and a default of two seconds for the converse second data delay. The delays may be needed to give the VRU time to invoke an application and allocate a touch-tone receiver to receive the passed digits.

If <data_1> is not "none," the converse first data delay timer starts when the call is answered by the VRU. Once the timer expires, the data_1 digits are outpulsed in-band to the VRU, followed by the end-of-string pound sign (#).

If <data_2> is not "none," the converse second data delay timer starts when the end-of-string pound sign (#) from the first digit string is outpulsed. Once the timer expires, the data_2 digits are outpulsed in-band to the VRU, followed by the end-of-string pound sign (#).

No time delays are invoked when the keyword "none" is administered.

Note:

The outpulsing of digits is not heard by the caller.

If the VRU hangs up during the data passing phase, the switch will log a vector event, reactivate vector processing at the next vector step, and ensure the VRU port is accessible for future calls.

Once all digits have been passed to the VRU, any audible feedback is disconnected.

Note:

At this point, control has effectively been passed to the VRU.

To ensure the robust operation of the VRU data passing operation, be sure to implement the following recommendations:

- Include the prompt and collect command in the VRU script for each data field passed in the **`converse-on`** step.
- Administer each prompt and collect command to recognize the "#" character as the end-of-string character.
- Ensure the number of digits expected is one greater than the number of digits passed to allow for the "#" character, which terminates every converse data field.

  Also, ensure no announcement is played in these prompt and collect steps.

- Ensure the first digit timeout in the prompt and collect steps is five seconds greater than the corresponding converse data delay. (For example, if the **converse-on** step passes two data fields, and if the converse first data delay is 0 secs and the converse second data delay is 4 secs, the first digit timeouts for the two prompt and collect commands should be at least 5 and 9 seconds, respectively.)
- Ensure the interdigit timeout in the prompt and collect steps is at least five seconds.
- Administer the converse first data delay to give a VRU under a heavy load sufficient time to allocate a DTMF touch-tone receiver after answering the call.
- Administer the converse second data delay to give a VRU under a heavy load sufficient time to complete any tasks between the first and second prompt and collect command. (For example, the VRU can invoke a new application if the first data field passed is used to identify the application script to be executed.)
- In general, for **converse-on** steps pass data to the VRU, ensure the VRU script does not execute any commands between the time the call is answered and the time when the first prompt and collect command is executed.

VRU data collection

When digits are passed from the switch to the VRU, the first VRU script commands executed are answer phone and prompt and collect. No announcement is programmed for the prompt and collect command, and the pound sign (#) is programmed as the end-of-string sign. If two sets of digits (that is, <data_1> and <data_2>) are passed by the switch, there will be two prompt and collect commands on the VRU to receive them.

If the first digit string (<data_1>) passed to the VRU is for application selection, the Conversant Script Builder exec command invokes the appropriate script. If a second digit string (<data_2>) is also used to pass an argument to this selected application, the first command in the executed script is a prompt and collect command with no announcement prompt programmed and with the pound sign (#) programmed as the end-of-string character.

The “Converse second data delay” is used to give the VRU time to invoke the selected application before the <data_2> digit string is outpulsed.

The application developer should ensure the administered `converse first data delay` and `converse second data delay` timers allow sufficient time for the VRU to successfully collect all outpulsed digits, even during periods of heavy call volume. Loss of digits from <data_2> is an indication the converse second data delay timer needs to be increased.

Script execution

During script execution, digits input by the calling party in response to prompt and collect commands are collected by the VRU but are not collected by the switch as dial-ahead digits. Also, audible feedback is determined by the VRU.

If an agent from a nonconverse split becomes available to service the call while the VRU script is being executed, the VRU port is dropped from the call, and the caller is immediately connected to the agent. Any digits collected prior to executing the **`converse-on`** step are still available and may be displayed using the CALLR-INFO button.

The entire call is dropped if the caller abandons during the execution of a **`converse-on`** step.

Data return

This phase is optional and is in effect only if the application calls for the VRU to return information to the switch before returning control to vector processing.

Digits returned by the VRU are treated as dial-ahead digits. The rules for collecting and processing VRU-returned digits are identical to those for collecting and processing Call Prompting digits (see Call Prompting on page 181).

VRU data return is done in a manner similar to an analog transfer. Specifically, the VRU does an analog switchhook flash, outpulses DTMF digits, and then hangs up. If converse data is returned, the DTMF digits comprise two parts. The first sequence of digits is the converse data return feature access code administered on the Feature-Access-Codes form. The second sequence of digits is the sequence to be passed by the VRU. These digits are collected later during vector processing.

The Conversant VRU offers a built-in external function called `converse_data`. This function allows applications developers to perform this operation in a convenient and robust fashion.

To ensure the robust operation of the VRU data return operation, be sure to follow these recommendations:

- Set the analog flash timing to 600 msecs.
- Ensure DTMF tones last at least 70 msec and interdigit pauses last at least 50 msec. This results in an outpulsing rate up to 8.33 digits per second.

- (Conversant only) Use the `converse_data` external function to return data to the switch.
- Hang up line to switch after outpulsing digits. Assume that switch will wait between 1.2 and 1.5 secs to determine that the hang-up is a disconnect.

For applications involving VRUs other than Conversant VRUs, be sure to follow these recommendations:

- After the flash, ensure the VRU performs dialtone detection (stutter dialtone) for a sufficient period of time to ensure accurate detection (typically 0.6 to 1.0 secs) before outpulsing the converse data return feature access code.
- If no dialtone is received before the timeout, ensure the VRU does two more retries of the analog flash. Also, if no dialtone is detected after two retries, ensure the VRU logs an error.
- Whenever dialtone is detected, ensure the digits of the converse data return feature access code are outpulsed.
- After the converse data return feature access code is outpulsed, the returned digits can be outpulsed without waiting for the second dial tone.
- After the VRU digits are outpulsed, the line to the switch is dropped.

Assuming an outpulse rate of 8 digits per sec (0.125 secs per digit), a 3-digit feature access code and stutter dial tone detection time of 0.6 secs, the maximum of 24 digits passed to switch should take about 6 secs (1.2 secs disconnect plus 8 secs plus 0.125 secs per digit).

The Call Classifiers required by the Call Prompting feature are not required for returning digits in-band from the VRU to the switch. Instead, general purpose TTR boards are used. As long as dial-ahead digits are available, any **`collect digits`** steps following a **`converse-on`** step do not require a Call Classifier to be allocated to the call.

If no general purpose TTRs are immediately available, and if the call queues for a TTR, no dial tone is provided. For this scenario, the VRU does not outpulse any digits until a TTR is available and dial tone is provided.

If there are no general purpose TTRs available on the switch, and if there is no space in the TTR queue, the operation fails. Usually, the VRU logs an error and then quits, and vector processing continues at the next vector step. Existing system measurements reports indicate when the system is configured with an insufficient number of TTRs.

The “Converse Data Return Code” can be followed by a maximum of 24 digits. The VRU touch-tones the code and the digits in-band. However, the code and the digits are not heard by the caller. The digits are stored in the switch as Call Prompting dial-ahead digits. If “x” digits are collected by vector processing before the **`converse-on`** step is executed, the maximum number of digits that can be returned is reduced to “24-x.” Any additional digits returned by the VRU are discarded. The data return is completed once the VRU hangs up.

The digit string returned by the VRU can consist of the digits (0 through 9) and pound signs (#). The pound sign (#) is interpreted by the **`collect digits`** step as an end-of-string character. If the digit string being returned is of variable length, the VRU can terminate the string with a pound sign (#) to avoid the ten second timeout delay that occurs when the digits are collected. If the digit string being returned is “multi-part” (that is, to be collected by multiple **`collect digits`** steps), and if some of the parts are of variable length, the pound sign (#) can be used to terminate each of the variable length parts.

Note:

An asterisk (*) may be included as part of the converse data return code. However, since the asterisk is interpreted as a “delete” character by the switch, it makes little sense to use it as a returned digit. If it is used as such, all characters returned prior to the asterisk are discarded.

During the data return phase, the caller is temporarily put on hold. Music-on-hold, if administered, is suppressed. Since the caller hears silence during this phase, feedback should be provided to the caller as soon as possible after the **`converse-on`** step is executed.

Any touch-tone digits dialed by the calling party during the data return phase are discarded. These digits do not cause data corruption, and they are not collected as dial-ahead digits by the switch.

If an interdigit timeout occurs during the data return phase, the switch logs a vector event, keeps the digits already returned, drops the VRU, and reactivates vector processing at the next vector step.

If the timeout occurs before the converse data return code is returned, the operation is the same except that no discarded digits will be available.

Script completion

The VRU script returns control to vector processing on the switch by simply hanging up the line. In cases where no data is returned to the switch, this is done usually by executing the **`quit`** command. In cases where data is returned, this occurs whenever the VRU hangs up on completion of the VRU data return operation.

The last set of digits collected before the **`converse-on split`** step is executed is still available and may be displayed by an answering agent on the nonconverse split by using the CALLR-INFO button.

A VRU script can be programmed to continue running after hanging up the voice line. This after-call work is usually very short, and it may involve either a final message to a host or a final update to a local database. For this scenario, the VRU port (channel) is still associated with the running script even though there is no longer a voice connection.

From the switch point of view, the agent (port) is available for the next call. If a call is delivered to this port, the VRU does not answer the call until the previous script has completed. As long as the VRU script's after call work is short in duration, this poses no significant problem for the VRI feature. However, high volume VRI applications with lengthy after call work periods should be avoided, especially if such periods are so lengthy they approach the administered timeout period on the switch for the Redirection on No Answer (RONA) feature. In such a case, RONA might think the VRU ports are faulty and might therefore start to take these ports out of service.

Switch data collection

This phase is in effect only if the VRU returns information to the switch.

Once the VRU script has completed and vector processing is reactivated, the returned digits are collected and processed by vector commands in the usual manner. Since the digits must be collected by a **`collect digits`** command, data may be returned and processed only if the Call Prompting option is enabled.

The data returned can consist of multiple parts. For example, the VRU could return a stream of seven digits in which a single digit success/fail code is followed by a six-digit account code. For this scenario, the **`converse-on`** step would be followed by a sequence of vector steps including two **`collect digits`** steps. The first **`collect digits`** step would collect one digit and then check the result code; the second **`collect digits`** step would collect the six-digit account code.

Any touch-tone digits dialed by the calling party during the data collection phase are discarded, do not cause data corruption, and are not collected as dial-ahead digits by the switch.

If VRU data is returned, the calling party is able to touch-tone a response to a switch prompt only after the data collection phase is completed and another **`collect digits`** step is executed. This is true because each executed **`collect digits`** step does not allocate a TTR when dial-ahead digits are present. Since VRU-returned digits are treated as dial-ahead digits, a TTR is attached to the call only after all returned digits are collected and another **`collect digits`** step is encountered. Only at this point can the caller hear an announcement for the **`collect digits`** command and successfully enter digits.

Appendix K: Security issues

Call Vectoring can be integrated into the security of your switch. For example, Call Vectoring and Call Prompting can be used to help prevent unauthorized users from gaining access to the switch via the Remote Access feature. This appendix explains how this is done.

Remote access

Abuse of remote access on the switch is one of the main methods by which unauthorized users obtain telephone services illegally. This section explains how a number of Call Vectoring features can be used to prevent unauthorized use of the remote access feature. No new development is required for any of these services.

Two methods are available, as follows:

- Front-ending remote access (that is, reaching the remote access extension via Call Vectoring).
- Replacing the function of the remote access extension by one or more call vectors.

Front-ending remote access

This method gives authorized external callers a VDN extension to call instead of the remote access extension, which is kept private. The corresponding call vector can then implement a number of security checks before routing callers to the remote access extension. Routing can be done via a **`route-to number`** or **`route-to digits`** step.

The following advantages are possible via this method:

- Call Vectoring can introduce a delay before the dial-tone is provided to the caller. Immediate dial-tone is often one criterion searched for by a hacker’s programs when the hacker is trying to break into a system.
- A recorded announcement declaring that the use of the switch services by unauthorized callers is illegal and that the call is subject to monitoring and/or recording can be played for the caller.
- Call Prompting can be used to prompt for a password. In such a case, the call is routed only if there is a match on the password.
- Use of the remote access extension can be limited to certain times of the day or certain days of the week.

- Real-time and historical reports on the use of the remote access feature can be accessed from CMS or from BCMS.
- Different passwords can be used on different days of the week or at different times during the day.
- Many VDNs that call the remote access extension can be identified. Accordingly, individuals or groups can be given their own VDN with unique passwords, permissions and reports. Any abuse of the system or security leak can then be attributed to an individual or a group.
- The caller can be routed to a VRU using the **`converse-on`** step where more sophisticated security checking, such as speaker recognition, can take place.
- Anyone failing any of the security checks can be routed to a “security” VDN that routes the caller to security personnel with a display set or to a VRU. Such a call would show “security” and possibly also the attempted password on the display. If the call is passed to a VRU, the VDN, the ANI and/or the prompted digits can be captured. CMS and BCMS reports on this security violation VDN will give information on how often and when security violations occur.

Replacing remote access

For this method, the remote access extension is not used. One or more VDNs are designed to access call vectors that can employ all of the security checks described in the previous section. The same reports and monitoring/recording capabilities described in the previous section can also be used. Instead of routing to the remote access extension, the vector collects digits from the caller and then routes to the given destination if there is a match on the password.

Again, multiple VDNs can be created for individuals or groups with different security checks and different permissions and/or restrictions. Destination numbers provided by callers can be screened by the vectors and denied if the user does not have permission to access that destination. For example, an individual user could be restricted to placing calls to numbers beginning with area codes “303” and “908.”

EAS

With EAS, agent stations can be locked when they are not staffed. This is accomplished by assigning the station a Class of Restriction that does not allow outbound calls or it could be restricted from toll calls.

EAS agents have an optional password of up to nine digits to log in. This password is not displayed on DCP terminals when the agent is entering the password on the dial pad.

Limiting outside access using VDN COR restrictions

Routing calls through the switch with Call Vectoring can raise some security issues. For more information on security issues, refer to the BCS Products Security Handbook.

A VDN has a Class Of Restriction (COR). Calls processed by the vector carry the permissions and restrictions associated with the COR of the VDN.

For example, if a vector in the switch is written to collect digits, and then to route to the digits dialed, the restrictions on what calls can be placed are determined by the COR of the latest VDN. Also, checks can be made on the digits that are dialed, using **`goto if digits`** vector commands (for example, **`goto if digits = 123`**) to disallow routing to undesired destinations. The collect digits step can also be limited to collect only the number of digits required (for example, only collecting five digits for internal dialing).

An incoming caller can access Trunk Access Codes, some Feature Access Codes, or most other sets of dialed digits. To deny incoming callers access to outgoing facility paths, the COR of the Vector Directory Number must be configured to disallow outgoing access. This should include the following: lowering the Facility Restriction Level (FRL) in the COR to the lowest acceptable value (FRL=0 provides the most restricted access to network routing preferences), assigning a Calling Party Restriction of “Toll” or “Outward” denying Facility Test Call capability, and blocking access to specific CORs assigned to outgoing Trunk Groups using the Calling Permissions section of the Class Of Restriction screen.

Review the Classes of Restriction assigned to your VDNs. If they are not restricted, consider assigning restrictions on the VDN and/or using **`goto`** tests on those digits to prevent callers from exiting the system via the vector.

Vector initiated service observing

The following restrictions can be used with vector initiated Service Observing to guard against unauthorized use.

- Call prompting commands can be used in Service Observing vectors to provide passcode protection, and to limit access to observing specific destinations or verified caller entered digits.
- Time of Day/Day of Week checks can be incorporated in Service Observing vectors.
- A vector can be created to be used exclusively for Service Observing.
- For a VDN to be observed as the result of a route-to command, the VDN must have a COR that allows it to be observed.
- The calling permissions of the COR assigned to the Service Observing VDN in conjunction with the “can be observed” settings of the COR assigned to the destination determine what agents, stations, or VDNS can be observed.

Voice response integration

When a converse step is used to access a VRU application that returns data for a collect digits step, the opportunity for toll fraud exists when the VRU application fails to return any data. To avoid this type of toll fraud be certain that one of the following is true:

- If the collected digits are used to route calls internally, be certain that the Class of Restriction (COR) for the Vector Directory Number (VDN) does not allow calls to route externally.
- If it is necessary to use the collected digits to route calls externally, use a password to verify that the collected digits have been passed by the VRU application. In the following vector example the VRU application returns a three-digit password followed by the eight-digit external number. The vector routes calls without the correct password to a different vector and routes calls with the correct password to the collected digits.

Voice Response Integration Security Example

```
converse-on split 10 pri m passing none and none
collect 3 digits after announcement none
goto vector 23 if digits <> 234
collect 8 digits after announcement none
route-to digits with coverage n
```

Attendant Vectoring

Security Violation Notification (SVN) referral calls can be directed to an attendant group. These are priority calls and, as such, cannot terminate to a VDN. However, when these calls are sent to the attendant group, they are treated as ordinary calls - priority does not apply to attendant group processing. So, these will be treated as normal attendant group calls and will be sent through vector processing.

Remote logout of agent

See Remote access on page 595 for issues associated with accessing the switch from a remote location.

Appendix L: Setting up a call center

Call center managers need some key indicators to measure ACD performance at their site. Usually, in setting up a call center, several factors involving call management are considered. The following list identifies and defines the most common of these factors, and it provides a typical question that might be asked. In addition, an insurance company example will be used to discuss the different options in this appendix.

Volume – Number of calls going in or out of the ACD. (How many calls did Split 1 answer?)

Productivity – Call volume per unit of time. (How many calls did Split 1 answer between 8 a.m. and 9 a.m.?)

Utilization – Overall use of the phone center. (What was my agent occupancy?)

Accessibility: – Availability of lines and agents when customers call the ACD (this is an area that the Avaya CMS can probably most clearly define and help improve). (Were lines busy when customers called or did they have to wait too long?)

Quality of Service: – Accuracy of information, a pleasant manner, responsiveness to caller concerns, successful completion of business, and efficient time utilization (not all measured directly by the CMS). (Was the caller given good service?)

This chapter explains how to set up a call center for customers with Call Vectoring and/or Expert Agent Selection (EAS).

Call Vectoring/non-EAS option

To set up a call center that has Call Vectoring but not EAS, do the following:

1. Determine your call center's objectives. Think about how you want your call center to handle calls and also about what you want your call center to achieve. See Non-EAS Worksheet #1: Call center objectives on page 604.

   A company's basic goals are to increase profits and market share and to decrease costs. The purpose of setting up a call center is to monitor these goals using the CMS/BCMS reports. It is best to have more than one objective. (Some customers set and then live by only one objective.) Call center objectives must then be created to meet the goals. These objectives must be communicated to the Split Supervisor or to the Administrator managing the call center.

   The following list provides an example set of call center objectives:

   - Establish the following measured entities:
     - Average Speed of Answer = 15 seconds
     - Abandon Rate <= 3%
     - Average Talk Time = 2 1/2 minutes
     - ACD calls per agent = 80 to 90 per day
     - Number of calls in queue = 6
     - Percentage of calls answered within the service level = 95%
     - Agent occupancy > 90%
     - Percentage of trunks busy < 3%
   - Generate revenue through the call center.
   - Train agents to back up each other.
   - Adequately train agents to provide service that meets customer expectations.

2. Review your existing operation and determine your call center needs (see Non-EAS Worksheet #2: Current split operation on page 605 and Customer/call center needs guidelines on page 601).

3. On the switch, assign a unique Hunt Group number and Call Distribution method to each caller need. This number will be your split number (see Non-EAS Worksheet #3: Customer needs on page 606 and Customer/call center needs guidelines on page 601).

4. Assign DNIS (Dialed Number Identification Service) (that is, the number dialed) as a Vector Directory Number (VDN) (see Customer/call center needs guidelines on page 601).

   As an option, you can assign one VDN for a main number and use Call Prompting to route the call to the proper split.

   The following table illustrates the guidelines given up to this point.

Customer/call center needs guidelines

Need	Split number (hunt group)	Call distribution[1]	VDN
New policy	1	UCD	555-6543
Questions about policy, Rate Quotes, Billing	2	UCD	555-6432
Spanish speaking for policy, service, and claims	3	DDC	555-6321
Claims	4	UCD	555-6210

1. Options include Direct Department Calling (DDC) and Uniform Call Distribution (UCD).

   Notice that this call center has only one split for all Spanish calls. However, resources permitting, you could create a New Policy split, a Service split, and a Claims split, each containing agents who speak Spanish. As an alternative, you could use one main VDN to point to a Call Prompting vector designed to route the calls to the splits.

5. On the switch, assign extensions to the agents' physical terminal locations (see the Extension/LoginID assignments table, below).

6. In CMS: Dictionary: Login Identifications, assign each agent a unique loginID. Agents are known to the CMS by the login ID. If assigned, reports refer to an agent by name, not by login ID.

   The following table illustrates the assignments described in the previous items:

Extension/LoginID assignments

Agent name[1]	Extension	LoginID[1]
Randy Tyler	1231	2000
Cathy Smith	1232	2001
Carla Silva	1238	2002

1. = assigned in the CMS Dictionary

Note:

When you are adding names to extensions on the switch, the agent name should be the same name as the loginID assigned in the CMS.

7. On the switch, assign agent extensions to splits. More than four splits can be assigned to an agent; however, the agent can log into a maximum of four splits. An agent assignment to splits can be changed in the CMS: ACD Administration: Move Extensions Between Splits if the agent is logged off.

   The following table illustrates the assignment of agent extensions to splits:

Agent extension/split assignments

Split (hunt group)	Agent extensions
1 - Sales	1231, 1232, 1233, 1234, 1235, 1236, 1237, 1238, 1239
2 - Service	1231, 1232, 1234, 1238, 1239, 1240
3 - Spanish	1238, 1240, 1245
4 - Claims	1238, 1239, 1240, 1241, 1242

8. On the switch or in the CMS: ACD Administration: VDN Assignments, assign a vector to each VDN. A VDN can point to only one vector. However, a vector can have more than one VDN pointing to it.

   The following table illustrates VDN/vector assignments.

VDN/vector assignments

VDN	Vector
6543	1 (Sales)
6432	2 (Service)
6321	3 (Spanish)
6210	4 (Claims)

9. On the switch or in the CMS: ACD Administration: Vector Contents, write your vectors. See Non-EAS Worksheet #4: Vector design on page 607.

   Your vectors should match your call center objectives. To meet these objectives, you must make a number of relevant decisions (for example, you may decide how soon you want to enlarge an agent pool or what kind of treatment the caller should receive). If your VDN and vector reports do not satisfy your call center objectives, you must consider your alternatives (for example, you may deem it necessary to train agents or to increase the amount of time elapsed from when a call queues to one split and then to another split).

The following lists indicate the actions produced by two different vectors:

Actions Produced by Vector #1:

a. Tell the caller to select one of the following prompts:
   - 1 = Sales
   - 2 = Service
   - 3 = Spanish
   - 4 = Claims
   - Nothing or 0 = Service

b. Queue the call.

c. Provide an announcement to the caller.

10. **Actions Produced by Vector #2:**

a. Queue the call to the correct service at a medium priority.

b. If no agents are available, provide a message and then play music.

c. If the call is not answered within 10 seconds, provide a second message and then play music.

d. If the call is not answered within 7 more seconds, queue the call to the Service split.

e. If the call is not answered within 7 more seconds, queue the call to the Spanish split at a high priority.

Note:

A **`check split`** command queues the call to up to three splits if the conditions are met. If the conditions are not met, the **`check split`** command may not get read again (if the vector step in which it appears is not executed again).

11. In the CMS: Dictionary, assign names to the splits, VDNs, and vectors.

12. Once your system is up and operational, you will need to monitor it to ensure you are meeting your call center objectives. The call management system can be used to monitor many of your objectives. Some objectives will need to be monitored and have adjustments made in real time. For example, if the number of calls waiting, average speed of answer, or percent answered within a service level is not meeting your objectives, you might want to immediately move some agents, direct calls to another vector, or look-ahead interflow some calls. Other items such as agent occupancy and percent all trunks busy may only need to be monitored daily to look for trends.

Non-EAS Worksheet #1: Call center objectives

What are my call center objectives?

Non-EAS Worksheet #2: Current split operation

Split ____________________

Primary Backup ________________

Secondary Backup ________________

Tertiary Backup ________________

List your customer/caller needs and your agent knowledge levels for this split.

1. ______________________________
2. ______________________________
3. ______________________________
4. ______________________________
5. ______________________________
6. ______________________________

Split ____________________

Primary Backup ________________

Secondary Backup ________________

Tertiary Backup ________________

List your customer/caller needs and your agent knowledge levels for this split.

1. ______________________________
2. ______________________________
3. ______________________________
4. ______________________________
5. ______________________________
6. ______________________________

Non-EAS Worksheet #3: Customer needs

Call center needs	Split number (hunt group)	Call distribution	VDN

Non-EAS Worksheet #4: Vector design

Vector #	Name:	Description:
Assigned VDNs:		
Assigned Trunk Groups:		
1.		
2.		
3.		
4.		
5.		
6		
7.		
8.		
9.		
10.		
11.		
12.		
13.		
14.		
15.		
16.		
17.		
18.		
19.		
20.		
21.		
22.		

23.
24.
25.
26.
27.
28.
29.
30.
31.
32.

EAS Worksheet #1: Call center objectives

What are my call center objectives?

EAS Worksheet #2: Current split operation

Split: ________________

Primary backup:		**Secondary backup:**		**Tertiary backup:**	
	______________		______________		______________

Customer/caller needs and agent skill sets in this split:	**Is agent expertise available? (y/n)**	**Do you want to separate skill sets with EAS? (y/n)**
1. ____________________	________	________
2. ____________________	________	________
3. ____________________	________	________
4. ____________________	________	________
5. ____________________	________	________
6. ____________________	________	________

Split: ________________

Primary backup:		**Secondary backup:**		**Tertiary backup:**	
	______________		______________		______________

Customer/caller needs and agent skill sets in this split:	**Is agent expertise available? (y/n)**	**Do you want to separate skill set with EAS? (y/n)**
1. ____________________	________	________
2. ____________________	________	________
3. ____________________	________	________
4. ____________________	________	________
5. ____________________	________	________
6. ____________________	________	________

EAS Worksheet #3: Customer needs

Call center needs	Split name	Split number (hunt group)	Call distribution	VDN

EAS Worksheet #4: Individual Agent Skills

				Logged-in coverage			Logged-out coverage		
Agent name	Login ID	Direct agent?	COR[1]	Point 1	Point 2	Point 3	Point 1	Point 2	Point 3

1. Class of restriction

EAS Worksheet #5: Agent Skills

Agent Name	Login ID	1st skill	2nd skill	3rd skill	4th skill	COR[1]

1. Class of restriction

EAS Worksheet #6: VDN Skill Preferences

VDN extension	VDN Name	COR[1]	Skill preferences			Vector
			1st skill	2nd skill	3rd skill	

1. Class of restriction

EAS Worksheet #7: Vector Design

Vector #	Name:	Description:
Assigned VDNs:		
Assigned Trunk Groups :		
1.		
2.		
3.		
4.		
5.		
6		
7.		
8.		
9.		
10.		
11.		
12.		
13.		
14.		
15.		
16.		
17.		
18.		
19.		
20.		
21.		
22.		

23.
24.
25.
26.
27.
28.
29.
30.
31.
32.

Appendix M: Converting a Call Center to EAS

The procedures in this Appendix provide guidelines for upgrading a call center from a non-EAS ACD environment to an EAS ACD environment. The primary activities involved in this conversion are:

- Step 1: Pre-EAS cutover administration for the system on page 617
- Step 2: Pre-EAS cutover administration for the CMS on page 622
- Step 3: Pre-EAS cutover administration for AUDIX on page 622
- Step 4: Pre-EAS cutover administration for ASAI on page 623
- Step 5: EAS cutover on page 623

Before the transition to EAS takes place, decisions must be made concerning:

- Which area of the current dial plan is to be used for EAS agent login IDs. EAS agent login IDs cannot conflict with already defined extension numbers (for example, an EAS agent login ID cannot be the same as a station extension number).
- Whether the current incoming call routing through VDNs and vectors will remain the same after the EAS upgrade, or whether new VDNs and/or vectors are required.
- How incoming call traffic is to be handled during EAS cutover.

Once these decisions are made, the pre-EAS cutover administration activities can be started in preparation for the conversion of the call center to EAS.

Note:

Even though EAS administration changes are being made, non-EAS ACD call handling and agent operations are unaffected. When the cutover to EAS is completed, all non-EAS ACD call handling and agent operations will cease.

Step 1: Pre-EAS cutover administration for the system

Perform the following activities to prepare the DEFINITY for the cutover to EAS:

1. At administration terminal display the System-Parameters Customer-Options form and verify that the `ACD`, `Expert Agent Selection`, and `Vectoring (Basic)` fields are set to y. If you will be using the increased capacities of `EAS-PHD`, verify that this option is set to y.
2. If you haven't already done so, display the Feature Access Code form and administer the ACD Agent Feature Access Codes (for example, "Login," "Logout," and "Auto-In") as required for call center agent operations.

3. Using the CDR System Parameters form, administer whether the EAS login ID, or the terminal extension where the EAS agent is logged in, should appear on CDR reports by setting the `Agent Login ID - Record` field to `y` or `n`. This field affects the CDR tracking for incoming calls only; outgoing calls made by a logged-in EAS agent are always recorded by CDR using the agent's login ID.
4. If new VDNs are desired for the EAS environment, using the VDN administration form, administer the VDN Skills and other VDN information for the VDNs used to route calls to EAS agents. If the "1st," "2nd," and/or "3rd" skill options are to be used in the vectors or for Avaya CMS tracking associated with these VDNs, then administer the 1st Skill, 2nd Skill, and 3rd Skill fields as required.
5. If new vectors are desired for the EAS environment, using the Vector administration form, administer the vectors associated with the VDNs added in the previous step. As part of the EAS feature, the "1st," "2nd," or "3rd" skill options may be used in the vector step fields where a skill hunt group is entered (rather than entering an absolute skill hunt group number). Refer to Expert Agent Selection on page 341 for more information concerning vector programming for the EAS feature.
6. If new skill hunt groups are required, using the Hunt Group administration form, administer the desired skill hunt groups.

   Note:

   Entering a **y** in the `Skills` field automatically causes the `ACD` and `Vector` fields to be set to `y`. With EAS optioned, it is not possible to administer members for a skill hunt group.
7. If coverage paths are to be administered for EAS agents, using the Coverage Path administration form, set up the coverage paths to be assigned to EAS agent login IDs.

   Note:

   There is a difference between coverage treatment for an EAS "Direct Agent" call (where both the calling party and called login ID have the `Direct Agent Calling COR` option set to `y`), and an EAS "personal" call (where either the calling party or called login ID does not have the `Direct Agent Calling COR` option set to `y`).

Note:

A Direct Agent call is routed to an EAS agent as an ACD-type call, and therefore its coverage behavior is considerably different from the coverage for a normal station call. For example, if an EAS agent is not available for an ACD call when a Direct Agent call is made to that agent, the Direct Agent call is queued to the Direct Agent Skill administered on the Agent Login ID form (after initiating a ring-ping and then fluttering the active work-mode button at the agent's terminal). On the other hand, a personal call to an EAS agent is not an ACD-type call, and its coverage behavior is similar to the coverage treatment for a call to a station extension. For example, a personal call to an EAS agent who is busy on any call appearance will result in the call being sent to an idle call appearance at that agent's terminal.

Depending on the type of coverage criteria desired for Direct Agent and personal calls to EAS login IDs, administer the desired coverage path criteria as follows:

- To provide coverage for a non-ACD "personal" call to an EAS login ID when the agent is logged in and active on any call appearance, set the `Active` coverage criteria to y. The `Active` coverage criteria does not apply for a Direct Agent call to an EAS login ID.
- To provide coverage for calls to an EAS login ID when the agent is logged out, set the `Busy` coverage criteria to y. Busy coverage will also be applied to a logged-in EAS agent when either of the following conditions occur:
  - — A Direct Agent call is made to the EAS agent and there are no available queue slots in the agent's first skill hunt group;
  - — A personal call is made to an EAS agent and the agent's station has no idle call appearances.
- To provide coverage for calls to an EAS login ID when the agent is logged in but does not answer after a certain number of ring cycles, set the `Don't Answer` coverage criteria to y, and enter a number for the desired ring time-out in the `Number of Rings` field.
- To provide immediate coverage for calls to an EAS login ID whether the agent is logged in or logged out, set the `All` coverage criteria to y.
- To provide coverage for calls to EAS login IDs when the call is to a logged-in agent who has activated the Send All Calls or Go To Cover features, set the `DND/SAC/Goto Cover` coverage criteria to y.

8. Up to three coverage paths for different types of call coverage criteria may be linked together by administering the Next Path Number field on the Coverage Path form. If the criteria for the first coverage path are not met, then the criteria for the second linked coverage path are checked by the system, and so on. This can be used to provide different coverage paths for calls to an EAS login ID when the associated agent is logged in or logged out.

   Note:

   If a call to a logged-in EAS login ID is a "personal" call and coverage goes into effect, the redirected call maintains a "simulated bridged appearance" at that agent's terminal. The agent may still answer the call after redirection takes place by going off-hook on this line appearance. However, if a call to a logged-in EAS login ID is a Direct Agent call, the redirected call does not maintain a simulated bridged appearance at the agent's terminal. The agent may not then answer the call after redirection takes place.

   Note:

   If the Redirection on No Answer (RONA) feature is enabled for skill hunt groups, set the ring time-out interval for the RONA feature such that it does not conflict with the coverage ring time-out criteria.

9. If coverage paths are administered for EAS login IDs, using the Feature-Related System Parameters form, set the Coverage - Subsequent Redirection No Answer Interval field to the desired ring time-out interval for calls routed to administered coverage points.

   Note:

   EAS login IDs may be administered as coverage points for a coverage path, and this administered coverage no-answer interval applies to Direct Agent or "personal" calls made to these coverage points as well.

10. Using the COR administration form, set the `Direct Agent Calling` field to y for any COR to be assigned to a trunk or station user who may initiate a Direct Agent call to an EAS agent, or to be assigned to an EAS login ID that may receive Direct Agent calls.

11. If EAS agent login ID passwords are to be administered, using the Feature-Related System Parameters form, set the `Minimum Agent-LoginID Password Length` field to the desired number of minimum password digits (0 to 9) which must be specified when agent passwords are administered via the Agent Login ID form. The total number of digits which may be assigned to a password is between the value of the `Minimum Agent-Login ID Password Length` field and 9 digits. If a password is administered for an agent, this password must be entered in addition to the agent's login ID to log in.

12. Using the Agent Login ID form, add the desired EAS login IDs to be associated with human agents, AUDIX ports, and/or AAS (Auto-Available Split) VRU ports. For human agents, the following fields are administered:
    - `Name`
    - `COR`
    - `Coverage Path` (optional)
    - `Security Code` (optional for Demand Print feature)
    - `LWC Reception` (optional)
    - `AUDIX Name` (for G3r only, if the `LWC Reception` field is set to `audix`, or if administered coverage path for the agent has an AUDIX coverage point)
    - `Password` (optional)
    - `Skills` - Skill Level (for at least one skill)
13. For AUDIX and AAS VRU port extensions, when these ports are associated with ACD-type hunt groups, these extensions must be associated with skill hunt groups as part of the cutover to EAS. Additionally, for skill hunt groups used for AAS ports, the AAS field must be set to y for these hunt groups before any EAS AAS agents can be administered.

    Note:

    AUDIX hunt groups do not need to be vector-controlled. This allows for ASAI monitoring of the skill hunt group.

    If AUDIX port extensions (such as for the Embedded AUDIX product) are not associated with an ACD hunt group, no administration is required for these ports as part of the cutover to EAS. For the AUDIX and/or AAS ports that are associated with ACD hunt groups, add EAS agent login IDs for these ports, where only the following fields need to be administered:
    - `Name`
    - `COR`
    - `Coverage path` (optional)
    - `AUDIX` (set to y for AUDIX ports)
    - AAS (set to y for AAS VRU ports)
    - `Port Extension` (set to the AUDIX or AAS port extension administered in the non-EAS environment)
    - `Skills` - Skill Level (where a single skill is entered for the skill hunt group associated with the AUDIX or AAS station ports)

14. Using the Station Administration form, administer any stations to be used by EAS agents and the desired work-mode buttons for each station (if not already administered).

Note:

If stations are already administered with work-mode buttons associated with splits, it is not necessary to readminister these buttons for EAS. If new work-mode buttons are added to a station, it is not possible to enter data in the Grp field after EAS is enabled except for the AUX work-mode button (which may be administered with a hunt group number if the entered hunt group is a non-ACD hunt group).

Note:

Also, if more than one set of work-mode buttons is administered on a station set, these buttons may be left as is until after the cutover to EAS. After the cutover, it is desirable to remove the extra sets of work-mode buttons since EAS requires only one set of work-mode buttons for agent operations.

Step 2: Pre-EAS cutover administration for the CMS

See *Avaya CMS Administration*, 585-215-515 for the procedures used to configure the CMS for the EAS feature. This document is also helpful in providing overall planning strategies for implementing call center operations.

Step 3: Pre-EAS cutover administration for AUDIX

If EAS agents' login IDs are administered with coverage paths that route to an AUDIX coverage point, the login IDs for these agents must be administered via the AUDIX console so that the caller will hear the appropriate AUDIX voice responses for calls made to EAS login IDs.

Note:

On the G3r, the `AUDIX Name` field on the Agent Login ID form must be set to the correct AUDIX name to provide proper AUDIX coverage of calls made to EAS agents, or to leave LWC messages for EAS agents if LWC reception to AUDIX is set up for the agents' login IDs.

Refer to Step 1: Pre-EAS cutover administration for the system for information on how to administer EAS login IDs for AUDIX port extensions on the DEFINITY.

Step 4: Pre-EAS cutover administration for ASAI

With ASAI-based applications for call center operations, the cutover to EAS may necessitate an upgrade of the ASAI-related application software on the adjunct. With OCM (Outgoing Call Management), the upgrade to EAS requires that specialized vectors be administered to handle the launching of calls from VDNs (as opposed to the non-EAS environment where OCM calls are launched from splits). For more information on the procedures to convert an ASAI application for EAS, see *Avaya MultiVantage CallVisor ASAI Technical Reference*, 555-230-220.

Step 5: EAS cutover

After all pre-EAS activities have been completed, the EAS feature may be activated. Just prior to the EAS cutover, a tape backup of the current DEFINITY translations should be made for possible recovery purposes in case some difficulty is encountered during cutover. In particular, since the transition to EAS results in the removal of all ACD hunt group members, the pre-EAS tape backup could save a considerable amount of time in restoring non-EAS hunt group translations if the cutover to EAS is not completed.

It is recommended that incoming ACD call traffic be blocked to prevent the queuing of new ACD calls to existing splits during the cutover from the non-EAS to EAS environment. Blocking of new incoming calls can be accomplished by:

- Busying out the appropriate trunk groups

  and/or

- Using the Vectoring form and setting the first vector step for actively used incoming call vectors to the “busy” step.

Once this is accomplished, perform these activities:

1. Make sure all EAS agents are logged out of all splits. If CMS or BCMS is operational, the CMS real-time reports for splits or the **`mon bcms split`** command can be used to identify the terminals where agents may still be logged in.
2. Issue the **`busy mis`** command at the administration terminal to busy-out the CMS link.
3. Issue the **`busy link n`** command at the administration terminal to busy-out any AUDIX switch-to-adjunct links.
4. Issue the **`busy station x`** command at the administration terminal to busy-out any AAS ports.
5. Using the Hunt Group form, convert any ACD splits to skill hunt groups by setting the `Skilled` field to **y** for these hunt groups.
6. Using the Feature-Related System Parameters form, set the `Expert Agent Selection (EAS) Enabled` field to **y**, and set the Adjunct CMS Release field.

7. Release the link to the CMS (if installed) by entering the **`release mis`** command at the administration terminal.
8. Inform the on-site agents that they can log into their terminals using the EAS login procedure and become available to receive ACD calls using the AUTO-IN or MANUAL-IN work-mode operations.
9. Using the Vectoring form, restore any vector steps temporarily changed to “busy” (to block incoming calls) to their previous vector step format.
10. Using the Trunk Group Administration form, if the routing for incoming trunks is to be changed to EAS-related VDNs, administer the `Incoming Destination` field for any trunk groups to the appropriate VDN extension number.
11. Issue the **`release station x`** command at the administration terminal to release any AAS ports (where the EAS login ID associated with each AAS port will be automatically logged in).
12. Issue the **`release link n`** command at the administration terminal to release any adjunct AUDIX links (where the adjunct will cause the associated ports to be logged in).

At this point, the cutover to EAS is complete. It is recommended that a backup of the switch translations be performed as soon as possible after the cutover to preserve the EAS-related administration changes. Also, if agent stations are administered with multiple sets of work-mode buttons, it is recommended that all but one set of work-mode buttons be removed from these stations. Also, multiple queue lights are required for EAS.

Appendix N: Feature Availability

This appendix lists available vectoring enhancements. For a detailed description of any item see the referenced section of this guide.

Vectoring (G3V4 Enhanced) provides the following additional capabilities:

- The ability to specify a priority level with the oldest-call-wait conditional on the *check* and *goto* commands. For more information about these commands, see Appendix A: Call Vectoring commands on page 387.
- The use of enhanced comparators (<>, >=, and <=) with the *goto* and *route-to* commands as well as use of “none” as an entry for digits checking, and “active” or “latest” VDN thresholds for indirect VDN references. See these commands in Appendix A: Call Vectoring commands on page 387.
- The use of the *interflow-qpos* conditional with the *goto* and *route-to* commands to achieve FIFO or FIFO-like call processing. See Look-Ahead Interflow (LAI) on page 203.
- The use of wildcards in digit strings for matching on collected digits and ANI or II-digits. See Appendix A: Call Vectoring commands on page 387.
- The use of Vector Routing Tables for matching on collected digits and ANI or II-digits. See Vector Routing Tables on page 188 or ANI /II-digits routing and Caller Information Forwarding (CINFO) on page 139.
- Multiple Audio/Music Sources for use with the *wait-time* command. See, Rolling Average Speed of Answer (ASA) on page 134.

Vectoring (G3V4 Advanced Routing) provides the following additional capabilities (Vectoring [G3V4 Enhanced] must also be enabled):

- Rolling Average Speed of Answer (ASA) Routing. See Rolling Average Speed of Answer (ASA) on page 134.
- Expected Wait Time (EWT) Routing. See Expected Wait Time (EWT) on page 126.
- VDN Calls Routing. See VDN Calls on page 137.

Vectoring (ANI/II-Digits Routing) provides the following additional capabilities (Vectoring [G3V4 Enhanced] must also be enabled):

- ANI Routing. See ANI /II-digits routing and Caller Information Forwarding (CINFO) on page 139.
- II-Digits Routing. See II-digits routing on page 144.

Vectoring (CINFO) provides the following additional capabilities (Call Prompting must also be enabled):

- The ability to collect ced and cdpd from the network. See Caller Information Forwarding on page 147.

Vectoring (Best Service Routing) automatically compares splits or skills in ACD environments to find the one that can provide the best service to each caller. BSR can operate at a single site, or it can be used with Look-Ahead Interflow to integrate a network of geographically distributed locations into a virtual call center. See Best Service Routing (BSR) on page 229.

Vectoring (Best Service Routing) without LAI enabled (single-site BSR) provides the following capabilities:

- The use of the *consider split/skill* command.
- The use of the *best* keyword with *queue-to*, *check*, and *goto* commands.
- The *wait-improved* conditional for *check* and *goto* commands. For a call that has already been queued, the *wait-improved* conditional gives you the ability to make any subsequent queuing conditional on the improvement in EWT as compared to the call's EWT in its current queue.

Vectoring (Best Service Routing) with LAI enabled (multi-site BSR) provides the following capabilities:

- The use of the *consider split/skill* and *consider location* commands.
- The use of the *reply-best* command to return data to the sending switch in response to a status poll.
- The use of the *best* keyword with *queue-to*, *check*, and *goto* commands.
- The *wait-improved* conditional for *check* and *goto* commands. For a call that has already been queued, the *wait-improved* conditional gives you the ability to make any subsequent queuing conditional on the improvement in EWT as compared to the call's EWT in its current queue.

Enhanced information forwarding provides the transport of existing call information and new call information such as Universal Call ID and Best Service Routing. See Information Forwarding on page 151.

Timed ACW provides the ability to assign a timed ACW interval to a VDN. See Vector Directory Number on page 47.

Appendix O: Improving performance

This appendix provides recommendations on how to write vectors that promote favorable performance practices. Improved performace depends on the following basic principles:

- Minimize the number of vector steps to process a call.
- Avoid vector steps which have a substantial probability of failure, such as:
  - Calls made outside of business hours
  - Queues to groups with less than desirable resources or characteristics.

The most wasteful use of processing resources is frequently caused by inefficient looping. For example, performance could be compromised when a vector loops through steps too often. This is especially true with long queue times.

Some examples with looping are discussed and recommendations are given on how to maximize performance. They are:

- Audible Feedback
- Look-Ahead Interflow
- Check

Examples other than looping are also discussed. They are:

- After Business Hours
- Look-Ahead Interflow

All looping examples in this appendix use only loops within a single vector. It is important to also be aware of looping to other vectors through the use of vector chaining. The same principles can be extrapolated from the looping examples. Creating a flow diagram is often helpful for identifying looping errors.

In addition to the example vectors, tables rating the relative performance costs of specific vector commands are also included.

Note:

Remember to test vectors for performance in addition to call flow.

Looping examples

Audible feedback

Recommendation: Evaluate the length of the wait period between repetitions of an announcement and increase the length, if possible. For optimum performance, add a second announcement after the initial announcement and repeat the second announcement less often.

The first example repeats the “All representative are busy. Please hold.” announcement every 10 seconds as long as the call is in queue.

Example: 10-second announcement interval

```
1. queue-to split 1
2. announcement 2770 (“All representatives are busy. Please hold.”)
3. wait-time 10 seconds hearing music
4. goto step 2 if unconditionally
5. stop
```

The next example repeats the announcement only every 60 seconds, thus improving performance.

Example: 60-second announcement interval

```
1. queue-to split 1
2. announcement 2770 (“All representatives are busy. Please hold.”)
3. wait-time 60 seconds hearing music
4. goto step 2 if unconditionally
5. stop
```

The next example adds a second announcement, “All representatives are still busy. Please hold.” in addition to the initial announcement and repeats the second announcement less often (every 120 seconds), thus improving performance again.

Example: Follow-up announcement

```
1. queue-to split 1
2. announcement 2770 (“All representatives are busy. Please hold.”)
3. wait-time 120 seconds hearing music
4. announcement 2771 (“All representatives are still busy. Please
                    continue to hold.”)
5. goto step 3 if unconditionally
6. stop
```

The following table compares the relative processing cost of the three examples by looking at the approximate number of vector steps executed while processing the call. Assumption is that the first announcement is 3 seconds long and the second announcement is 4 seconds long.

Approximate number of vector steps executed for the audible feedback examples

Initial conditions	**Example:** **10-second announcement interval**	**Example:** **60-second announcement interval**	**Example:** **Follow-up announcement**
An agent is available in split 1	1	1	1
Queueing time of 5 minutes	70	15	9

When a call is queued for 5 minutes, the number of vector steps drops dramatically when the amount of time between announcements is increased, and drops even more when a second announcement is added, and the amount of time between announcements is increased again. When an agent in split 1 is immediately available to answer the call, there is no difference in the number of vector steps for the three examples.

Look-Ahead interflow

Recommendation 1: Use the **`interflow-qpos`** conditional to achieve FIFO (first in, first out) or near-FIFO call processing. For more information, see Look-Ahead Interflow (LAI) on page 203.

Recommendation 2: If you do not have the **`interflow-qpos`** conditional, add a wait period between successive look-ahead interflow attempts and make the waiting period as long as feasible.

The following example continuously attempts a look-ahead interflow as long as the call is in queue or until a look-ahead attempt succeeds.

Example: continuous look ahead - no delay

```
1. queue-to split 1 pri l
2. announcement 3000
3. wait-time 20 seconds hearing music
4. route-to number 93035555555 cov n if unconditionally
5. goto step 4 if unconditionally
```

The example shown above adds a delay so that the look-ahead interflow attempt occurs only every 10 seconds.

Example: look ahead with 10 second delay

```
1. queue-to split 1 pri l
2. announcement 3000
3. wait-time 20 seconds hearing music
4. route-to number 93035555555 cov n if unconditionally
5. wait-time 10 seconds hearing music
6. goto step 4 if unconditionally
```

The next example increases performance even more by increasing the delay between look-ahead interflow attempts to 30 seconds.

Example: look ahead with 30 second delay

```
1. queue-to split 1 pri l
2. announcement 3000
3. wait-time 20 seconds hearing music
4. route-to number 93035555555 cov n if unconditionally
5. wait-time 30 seconds hearing music
6. goto step 4 if unconditionally
```

The following table compares the relative processing cost of the three examples by looking at the approximate number of vector steps executed while processing the call. Assumption is that the announcement is 5 seconds long.

Approximate number of vector steps executed for look-ahead interflow examples

Initial conditions	Example: look ahead with no delay	Example: look ahead with 10 second delay	Example: look ahead with 30 second delay
An agent is available in split 1	1	1	1
Queueing time of 5 minutes	up to 1,000	85	30

Check

Recommendation: When using check commands to queue a call to backup splits, ensure that an adequate amount of time has elapsed before checking the backup splits again.

Note:

With the Expected Time Wait Time feature, the style of programming used in this example is not optimal. The best approach is to use the Expected Time Wait feature to locate the most appropriate split for the call and queue it there.

The next example checks backup splits continuously as long as the call is in queue.

Example: Continuous check

```
1. queue-to split 1 pri h
2. announcement 3000
3. wait-time 10 seconds hearing music
4. check split 21 pri m if available-agents > 0
5. check split 22 pri m if available-agents > 0
6. check split 23 pri m if available-agents > 0
7. check split 24 pri m if available-agents > 0
8. check split 25 pri m if available-agents > 0
9. goto step 4 if unconditionally
```

The next example adds a delay of 10 seconds to ensure that some time has elapsed before checking the backup splits again.

Example: Check with 10 second delay

```
1. queue-to split 1 pri h
2. announcement 3000
3. wait-time 30 seconds hearing music
4. check split 21 pri m if available-agents > 0
5. check split 22 pri m if available-agents > 0
6. check split 23 pri m if available-agents > 0
7. check split 24 pri m if available-agents > 0
8. check split 25 pri m if available-agents > 0
9. wait-time 10 seconds hearing music
10. goto step 4 if unconditionally
```

Since the agent availability status may not be likely to change every 10 seconds, it may make sense to increase the wait time to 30 seconds, as shown in the example in The following example.

Example: Check with 30 second delay

```
1. queue-to split 1 pri h
2. announcement 3000
3. wait-time 30 seconds hearing music
4. check split 21 pri m if available-agents > 0
5. check split 22 pri m if available-agents > 0
6. check split 23 pri m if available-agents > 0
7. check split 24 pri m if available-agents > 0
8. check split 25 pri m if available-agents > 0
9. wait-time 30 seconds hearing music
10. goto step 4 if unconditionally
```

The following table compares the relative processing cost of the three examples by looking at the approximate number of vector steps executed while processing the call. Assumption is that the announcement is 5 seconds long.

Approximate number of vector steps executed for check examples

Initial conditions	Example: continuous check	Example: check with 10-second delay	Example: Check with 30-second delay
An agent is available in split 1	1	1	1
Queueing time of 5 minutes	up to 1,000	190	65

When a call is queued for 5 minutes, the number of vector steps drops dramatically when a delay is added before checking the backup splits again, and drops even more when the length of the delay is increased again. When an agent in split 1 is immediately available to answer the call, there is no difference in the number of vector steps for the three examples.

Other examples

After business hours

Recommendation: Test to see if the destination resources are available (such as during business hours) before queuing.

The following example queues calls to a hunt group regardless of the time of the call. When the call is made after business hours, the announcement is repeated until the caller hangs up.

Unconditional queuing to hunt group

```
1. queue-to split 1
2. announcement 5000
                   ("All agents are busy. Please hold.")
3. wait-time 120 seconds hearing music
4. announcement 5001
                   ("All agents are still busy. Please continue to
                   hold.")
5. goto step 3 if unconditionally
```

The next example tests for business hours before queuing the call. If the call is made after business hours, an announcement informs the caller of the business hours and the call is terminated.

Queue to hunt group with time-of-day conditional

```
1. goto step 7 if time-of-day is all 17:00 to all 8:00
2. queue-to split 1
3. announcement 5000
                              ("All agents are busy. Please hold.")
4. wait-time 120 seconds hearing music
5. announcement 5001
                              ("All agents are still busy. Please
                              continue to hold.")
6. goto step 4 if unconditionally
7. disconnect after announcement 5001
                              ("Business hours are 8:00 AM to 5:00
PM,
                              Please call back then.")
```

In the first example, unnecessary processing occurs when a call is queued after business hours and the call is terminated only when the caller hangs up. As shown in the second example, it is more economical to test for business hours before queuing a call.

Look-ahead interflows

Recommendation: When using a look-ahead interflow, first test to see if the receiving office is open for business.

The scenario is a sending switch in Los Angeles, with office hours from 8:00 AM to 5:00PM (8:00-17:00) PST and the receiving switch is in New York, with office hours from 8:00 AM to 5:00PM EST (5:00-14:00 PST). There is a three hour difference between the two switches

The following example routes calls to the New York switch. If there are no agents available at the Los Angeles switch, it is possible for calls to be interflowed during hours that the agents in New York are not available, thus doing unnecessary processing.

Unconditional Look-ahead interflow

```
1. queue-to split 1
2. route-to number 99145555555 cov n if unconditionally
3. announcement 2770 (“All agents are busy. Please hold.”)
4. wait-time 120 seconds hearing music
5. goto step 3 if unconditionally
6. stop
```

The next example tests first to see if the New York switch is open before requesting a queue to the New York switch, thus avoiding unnecessary processing.

Look-ahead interflow with time-of-day condition

```
1. queue-to split 1
2. goto step 4 if time-of-day is all 14:00 to all 05:00
3. route-to number 99145555555 cov n if unconditionally
4. announcement 2770                    (“All agents are busy.
Please hold.”)
5. wait-time 120 seconds hearing music
6. goto step 4 if unconditionally
7. stop
```

The next example can be used if you have Advanced Routing optioned. In this case, the Expected Wait Time feature may be used to determine whether it is worthwhile placing a look-ahead interflow call attempt.

Look-ahead interflow with expected wait time and time-of-day conditions

```
1. queue-to split 1
2. goto step 5 if expected-wait for call < 30
3. goto step 5 if time-of-day is all 14:00 to all 05:00
4. route-to number 99145555555 cov n if unconditionally
5. announcement 2770                    (“All agents are busy.
Please hold.”)
6. wait-time 120 seconds hearing music
7. goto step 5 if unconditionally
8. stop
```

In the examples shown above, note that there is no reason to attempt an interflow if the call will be answered quickly at the main switch. Therefore, vector steps that do not facilitate rapid call response are avoided.

Glossary

AAR	See Automatic Alternate Routing (AAR).
abandoned call	An incoming call in which the caller hangs up before the call is answered.
ACD	See Automatic Call Distribution (ACD).
ACD agent	See agent.
ACW	See after-call work (ACW) mode.
ACD	See Automatic Call Distribution (ACD). ACD also refers to a work state in which an agent is on an ACD call.
ACD work mode	See work mode.
active-notification association	A link that is initiated by an adjunct, allowing it to receive event reports for a specific switch entity, such as an outgoing call.
active-notification call	A call for which event reports are sent over an active-notification association (communication channel) to the adjunct. Sometimes referred to as a monitored call.
active notification domain	VDN or ACD split extension for which event notification has been requested.
adjunct	A processor that does one or more tasks for another processor and that is optional in the configuration of the other processor. See also application.
adjunct-controlled split	An ACD split that is administered to be under adjunct control. Agents logged into such splits must do all telephony work, ACD login/ logout, and changes of work mode through the adjunct (except for auto-available adjunct-controlled splits, whose agents may not log in/out or change work mode).
adjunct-monitored call	An adjunct-controlled call, active-notification call, or call that provides event reporting over a domain-control association.
Adjunct-Switch Application Interface (ASAI)	A recommendation for interfacing adjuncts and communications systems, based on the CCITT Q.932 specification for layer 3.
adjusted EWT	Expected Wait Time (EWT) plus a user adjustment set by a **`consider`** command.

after-call work (ACW) mode A mode in which agents are unavailable to receive ACD calls. Agents enter the ACW mode to perform ACD-related activities such as filling out a form after an ACD call.

AG ASAI Gateway

agent A person who receives calls directed to a split. A member of an ACD hunt group or ACD split. Also called an ACD agent.

agent report A report that provides historical traffic information for internally measured agents.

agent selection method The method the switch uses to select an agent in a hunt group when more than one agent is available to receive the next call: UCD-MIA, UCD-LOA, EAD-MIA, or EAD-LOA

ANI See Automatic Number Identification (ANI).

application An adjunct that requests and receives ASAI services or capabilities. One or more applications can reside on a single adjunct. However, the switch cannot distinguish among several applications residing on the same adjunct and treats the adjunct, and all resident applications, as a single application. The terms application and adjunct are used interchangeably throughout this document.

In Best Service Routing, an application is any specific implementation of multi-site Best Service Routing.

application plan Used only in multi-site BSR applications, the application plan identifies the remote switches that may be compared in consider series. The plan also specifies the information used to contact each switch and to interflow calls to it.

ARS See Automatic Route Selection (ARS).

ASAI See Adjunct-Switch Application Interface (ASAI)

auto-in trunk group Trunk group for which the CO processes all of the digits for an incoming call. When a CO seizes a trunk from an auto-in trunk group, the switch automatically connects the trunk to the destination — typically an ACD split where, if no agents are available, the call goes into a queue in which callers are answered in the order in which they arrive.

Auto-In Work mode One of four agent work modes: the mode in which an agent is ready to process another call as soon as the current call is completed.

Automatic Alternate Routing (AAR) A feature that routes calls to other than the first-choice route when facilities are unavailable.

Automatic Call Distribution (ACD)
A feature that answers calls, and then, depending on administered instructions, delivers messages appropriate for the caller and routes the call to an agent when one becomes available.

Automatic Call Distribution (ACD) split
A method of routing calls of a similar type among agents in a call center. Also, a group of extensions that are staffed by agents trained to handle a certain type of incoming call.

Automatic Number Identification (ANI)
Representation of the calling number, for display or for further use to access information about the caller. Available with Signaling System 7.

Automatic Route Selection (ARS)
A feature that allows the system to automatically choose the least-cost way to send a toll call.

Aux-Work mode
A work mode in which agents are unavailable to receive ACD calls. Agents enter Aux-Work mode when involved in non-ACD activities such as taking a break, going to lunch, or placing an outgoing call.

available agent strategy
Part of the Best Service Routing feature, the available agent strategy determines how BSR commands in a vector identify the best split or skill when several have available agents. The possible available agent strategies are: UCD-MIA, UCD-LOA, EAD-MIA, and EAD-LOA.

AWT
Average work time

BCMS
Basic Call Management System

best
The split, skill, or location that will provide the best service for a caller as determined by Best Service Routing.

Best Service Routing (BSR)
A DEFINITY feature, based on call vectoring, that routes ACD calls to the split, skill, or call center best able to service each call. Best Service Routing can be used on a single switch, or it can be used to integrate resources across a network of DEFINITY Enterprise Communication Servers.

BSR
See Best Service Routing (BSR).

CACR
Cancellation of Authorization Code Request

Call Management System (CMS)
An application, running on an adjunct processor, that collects information from an ACD unit. CMS enables customers to monitor and manage telemarketing centers by generating reports on the status of agents, splits, trunks, trunk groups, vectors, and VDNs, and enables customers to partially administer the ACD feature for a communications system.

call-reference value (CRV)
An identifier present in ISDN messages that associates a related sequence of messages. In ASAI, CRVs distinguish between associations.

call vector A set of up to 15 vector commands to be performed for an incoming or internal call.

call work code A number, up to 16 digits, entered by ACD agents to record the occurrence of customer-defined events (such as account codes, social security numbers, or phone numbers) on ACD calls.

CAS Centralized Attendant Service or Call Accounting System

cause value A value is returned in response to requests or in event reports when a denial or unexpected condition occurs. ASAI cause values fall into two coding standards: Coding Standard 0 includes any cause values that are part of AT&T and CCITT ISDN specifications; Coding standard 3 includes any other ASAI cause values. This document uses a notation for cause value where the coding standard for the cause is given first, then a slash, then the cause value. Example: CS0/100 is coding standard 0, cause value 100.

Class of Restriction (COR) A feature that allows up to 64 classes of call-origination and call-termination restrictions for telephone, telephone groups, data modules, and trunk groups. See also Class of Service (COS).

Class of Service (COS) A feature that uses a number to specify if voice-terminal users can activate the Automatic Callback, Call Forwarding All Calls, Data Privacy, or Priority Calling features. See also Class of Restriction (COR).

CMS Call Management System

consider series Consider commands are typically written in a set of two or more. This set of consider commands is called a consider series.

consider sequence A consider series plus a **`queue-to best`**, **`check-best`**, or **`reply-best`** step is called a consider sequence.

COR See Class of Restriction (COR).

COS See Class of Service (COS).

coverage answer group A group of up to eight telephones that ring simultaneously when a call is redirected to it by Call Coverage. Any one of the group can answer the call.

coverage call A call that is automatically redirected from the called party's extension to an alternate answering position when certain coverage criteria are met.

coverage path The order in which calls are redirected to alternate answering positions.

coverage point An extension or attendant group, VDN, or ACD split designated as an alternate answering position in a coverage path.

covering user	A person at a coverage point who answers a redirected call.
CWC	See call work code.
DAC	1. Dial access code or Direct Agent Calling
DCS	Distributed Communications System
DDC	Direct Department Calling
direct agent	A feature, accessed only via ASAI, that allows a call to be placed in a split queue but routed only to a specific agent in that split. The call receives normal ACD call treatment (for example, announcements) and is measured as an ACD call while ensuring that a particular agent answers.
distributed communications system (DCS)	A network configuration linking two or more communications systems in such a way that selected features appear to operate as if the network were one system.
DIVA	Data In/Voice Answer
DNIS	Dialed-Number Identification Service
domain	VDNs, ACD splits, and stations. The VDN domain is used for active-notification associations. The ACD-split domain is for active-notification associations and domain-control associations. The station domain is used for the domain-control associations.
domain-controlled split	A split for which `Third_Party_Domain_Control` request has been accepted. A domain-controlled split provides an event report for logout.
domain-controlled station	A station for which a `Third_Party_Domain_Control` request has been accepted. A domain-controlled station provides event reports for calls that are ringing, connected, or held at the station.
domain-controlled station on a call	A station that is active on a call, and which provides event reports over one or two domain-control associations.
ETN	Electronic tandem network
EWT	See expected wait time (EWT).
expected wait time (EWT)	Prediction of how long a call will wait in queue before it is answered.

extension-in	Extension-In (ExtIn) is the work state agents go into when they answer (receive) a non-ACD call. If the agent is in Manual-In or Auto-In and receives an extension-in call, it is recorded by CMS as an AUX-In call.
extension-out	The work state that agents go into when they place (originate) a non-ACD call.
external measurements	Those ACD measurements that are made by the External CMS adjunct.
extension	A number by which calls are routed through a communications system or, with a Uniform Dial Plan (UDP) or main-satellite dialing plan, through a private network.
FAC	Feature Access Code
hunt group	A group of extensions that are assigned the Station Hunting feature so that a call to a busy extension reroutes to an idle extension in the group. See also ACD work mode.
ICD	Inbound Call Director
ICDOS	International Customer-Dialed Operator Service
ICHT	Incoming call-handling table
ICI	Incoming call identifier
ICM	Inbound Call Management
interflow	To route an incoming call to an external switch without answering it at the origin switch.
intraflow	The ability for calls to redirect to other splits on the same PBX on a conditional or unconditional basis using call coverage busy, don't answer, or all criteria.
internal measurements	BCMS measurements that are made by the system. ACD measurements that are made external to the system (via External CMS) are referred to as external measurements.
LDN	Listed directory number
Manual-In work mode	One of four agent work modes: the mode in which an agent is ready to process another call manually. See Auto-In Work mode for a contrast.
message center	An answering service that supplies agents to and stores messages for later retrieval.
message center agent	A member of a message-center hunt group who takes and retrieves messages for voice-terminal users.

monitored call	See active-notification call.
OCM	Outbound Call Management
other split	The work state that indicates that an agent is currently active on another split's call, or in ACW for another split.
poll	See status poll.
poll suppression	A component of BSR intelligent polling that eliminates wasteful polling of remote locations which have returned poor adjusted EWTs.
polling, intelligent	An automatic feature of Best Service Routing that significantly reduces the number of status polls executed. When a remote location cannot be the best resource at a given moment in time, the intelligent polling feature temporarily suppresses polls to that location.
redirection criteria	Information administered for the coverage path of each telephone, that determines when an incoming call is redirected to coverage.
Redirection on No Answer	An optional feature that redirects an unanswered ringing ACD call after an administered number of rings. The call is then redirected back to the agent.
report scheduler	Software that is used in conjunction with the system printer to schedule the days of the week and time of day that the desired reports are to be printed.
split	See ACD work mode.
split condition	A condition whereby a caller is temporarily separated from a connection with an attendant. A split condition automatically occurs when the attendant, active on a call, presses the start button.
split number	The split's identity to the switch and BCMS.
split report	A report that provides historical traffic information for internally measured splits.
split (agent) status report	A report that provides real-time status and measurement data for internally measured agents and the split to which they are assigned.
staffed	Indicates that an agent position is logged in. A staffed agent functions in one of four work modes: Auto-In, Manual-In, ACW, or AUX-Work.
status poll	A call placed by a `consider location` vector command to obtain status data from a remote location in a multi-site BSR application.
stroke counts	A method used by ACD agents to record up to nine customer-defined events per call when CMS is active.

system report A report that provides historical traffic information for internally measured splits.

system-status report A report that provides real-time status information for internally measured splits.

to control An application can invoke Third Party Call Control capabilities using either an adjunct-control or domain-control association.

to monitor An application can receive event reports on an active-notification, adjunct-control, or domain-control association.

UCD Uniform call distribution

VDN See vector directory number (VDN).

vector directory number (VDN) An extension that provides access to the Vectoring feature on the switch. Vectoring allows a customer to specify the treatment of incoming calls based on the dialed number.

vector-controlled split A hunt group or ACD split administered with the vector field enabled. Access to such a split is possible only by dialing a VDN extension.

work mode One of four states (Auto-In, Manual-In, ACW, AUX-Work) that an ACD agent can be in. Upon logging in, an agent enters AUX-Work mode. To become available to receive ACD calls, the agent enters Auto-In or Manual-In mode. To do work associated with a completed ACD call, an agent enters ACW mode.

work state An ACD agent may be a member of up to three different splits. Each ACD agent continuously exhibits a work state for every split of which it is a member. Valid work states are Avail, Unstaffed, AUX-Work, ACW, ACD (answering an ACD call), ExtIn, ExtOut, and OtherSpl. An agent’s work state for a particular split may change for a variety of reasons (example: when a call is answered or abandoned, or the agent changes work modes). The BCMS feature monitors work states and uses this information to provide BCMS reports.

Index

Symbols

A

B

C

D

E

M

N

O

P

Q

R

S

T

U

V

W