NORTEL NETWORKS

Passport 7400, 15000, 20000

Management System User Interface Guide

241-5701-045

Passport 7400, 15000, 20000

Management System User Interface Guide

Publication: 241-5701-045
Document status: Standard
Document version: 5.2S1
Document date: November 2003

Printed in Canada

Publication history

November 2003

5.2S1 Standard

General availability. Contains information on Passport 7400, Passport 15000, and Passport 20000 for the PCR5.2 GA release.

Contents

List of figures

List of tables

About this document

This document describes the Passport management system user interface, how to access it, and how to use it to configure, operate and maintain the Passport system.

The following topics are discussed in this section:

- “Who should read this document and why” (page 11)
- “What you need to know” (page 11)
- “How this document is organized” (page 12)
- “What’s new in this document” (page 12)
- “Text conventions” (page 13)
- “Related documents” (page 13)
- “How to get more help” (page 14)

Who should read this document and why

This guide is for anyone who performs the following tasks for configuring the Passport system:

- planning
- installing and provisioning
- operating and maintaining

What you need to know

This guide assumes that you understand the architecture and operation of Passport products. You also require basic UNIX knowledge.

You can acquire Passport product knowledge by reading 241-5701-030 *Passport 7400, 15000, 20000 Overview*.

Before you operate and maintain Passport, make sure you understand the following:

- Passport concepts
    - Passport hardware and software
    - Passport installation, commissioning, and provisioning
    - Passport-to-Passport interworking
    - Passport-to-DPN-100 interworking (applicable to Passport 7400 series only)
- UNIX
    - UNIX workstations
    - UNIX operating system, its facilities, and commands
- standard network operations and maintenance activities
- Preside Multiservice Data Manager workstation concepts

How this document is organized

This document contains the following sections:

- “Passport management system” (page 15)
- “Accessing a Passport node from a network management workstation” (page 19)
- “Passport provisioning system” (page 31)

What’s new in this document

There were no new features added to this document.

Other changes made to this document include the following:

- Updated content throughout the NTP to reflect new MTBI standards, and address several technical issues.

Text conventions

This document uses the following text conventions:

- `nonproportional spaced plain type`

  Nonproportional spaced plain type represents system generated text or text that appears on your screen.

- **`nonproportional spaced bold type`**

  Nonproportional spaced bold type represents words that you should type or that you should select on the screen.

- *italics*

  Statements that appear in italics in a procedure explain the results of a particular step and appear immediately following the step.

  Words that appear in italics in text are for naming.

- `[optional_parameter]`

  Words in square brackets represent optional parameters. The command can be entered with or without the words in the square brackets.

- `<general_term>`

  Words in angle brackets represent variables which are to be replaced with specific values.

Related documents

See the following documents for related information:

- 241-5701-050 *Passport 7400, 15000, 20000 Commands*
- 241-5701-060 *Passport 7400, 15000, 20000 Components*
- 241-5701-270 *Passport 7400, 15000, 20000 Software Installation Guide*
- 241-5701-300 *Passport 7400, 15000, 20000 SNMP Guide*
- 241-5701-500 *Passport 6400, 7400, 15000, 20000 Alarms*

- 241-5701-520 *Passport 7400, 15000, 20000 Troubleshooting and Testing*
- 241-5701-600 *Passport 7400, 15000, 20000 Configuration Guide*
- NN10600-605 *Passport - MDM Network Security: Operations*
- 241-5701-611 *Passport 7400, 15000, 20000 Data Collection Guide*
- 241-5701-650 *Passport 7400, 15000, 20000 Accounting Fundamentals*
- 241-6001-023 *Preside MDM Configuration Management for Passport User Guide*
- 241-6001-801 *Preside MDM Overview*
- For information on last minute updates, see the Release Notes for the *current release*.
- *Engineering Notes and Guidelines* provide performance specifications for the relative product.

How to get more help

For information on training, problem reporting, and technical support, see the “Nortel Networks support services” section in the product overview document.

Chapter 1
Passport management system

The Passport network management system combines hardware and software to provide an interface to a network. The network management system uses that interface to control and monitor a single node or many nodes in the network.

There are five functions of network management. See the following sections for information on each function:

- “Fault management” (page 15)
- “Configuration management” (page 16)
- “Accounting management” (page 16)
- “Performance management” (page 16)
- “Security management” (page 16)

To perform these management functions, you must use a network management device. Passport provides a number of network management devices with both text and graphical interfaces. For more information, see “Network management devices” (page 16).

Fault management

The key to Passport fault management is alarms. An alarm indicates that there is a fault in your network. OSI state information can help you determine the cause of a fault. For information on OSI state information, see 241-5701-520

Passport 7400, 15000, 20000 Troubleshooting and Testing. For information on specific alarms, see 241-5701-500 *Passport 6400, 7400, 15000, 20000 Alarms*.

Configuration management

You perform the configuration management function using the Passport provisioning system. The provisioning system allows you to change the current configuration of your Passport node and to manage saved configurations. For information on the provisioning system, see "Passport provisioning system" (page 31). For information on configuring the base Passport system or managing the software running on your Passport node, see 241-5701-600 *Passport 7400, 15000, 20000 Configuration Guide*.

Accounting management

The Passport data collection system collects information about the usage of certain services. You can use this data for billing customers. For information about Passport accounting, see 241-5701-650 *Passport 7400, 15000, 20000 Accounting Fundamentals*.

Performance management

The Passport data collection system collects statistical and real-time statistical data. You can use this data to calculate the system's performance. For more information on Passport performance management, see 241-5701-611 *Passport 7400, 15000, 20000 Data Collection Guide*.

Security management

Passport provides various levels of security including user ID and password protection, command logging, an allowed-IP list, and session control. For more information on Passport security management, see NN10600-605 *Passport - MDM Network Security: Operations*.

Network management devices

You can manage your Passport node through the following management devices:

- "Local terminal" (page 17)
- "Telnet" (page 17)

- “Preside Multiservice Data Manager” (page 17)
- “Simple network management protocol” (page 17)

For information on the interfaces that support these network management devices, see “Network management interfaces” (page 22).

Local terminal

The most direct method for managing a Passport node is to use a VT100 terminal or a VT100 terminal emulator plugged directly into the V.24 port on any of the node’s control processors. This local access provides you with a text interface where you can enter commands and view alarms.

Telnet

With a local VT100 terminal, you must be in the same location as your Passport node to manage it. Often this is not possible or practical. You can manage your node from a remote site using a telnet application. With telnet, you can log into Passport’s text interface from a remote workstation. You can enter commands and view alarms as if you were using the local VT100 terminal.

Preside Multiservice Data Manager

For sophisticated network management capabilities and a graphical interface, you can use the Preside Multiservice Data Manager. The Preside Multiservice Data Manager, which connects to Passport nodes through the proprietary FMIP protocol, allows you to manage a Passport network as well as a combined Passport and DPN-100 network. For more information on Preside Multiservice Data Manager, see 241-6001-801 *Preside MDM Overview*.

Simple network management protocol

Passport supports the simple network management protocol (SNMP). This standard network management protocol lets you manage Passport with third-party network management systems, such as HP OpenView.

For information on SNMP, see 241-5701-300 *Passport 7400, 15000, 20000 SNMP Guide*.

Chapter 2
Accessing a Passport node from a network management workstation

To manipulate and view nodes in a network you must log into the network with a network management workstation. When you log into a Passport node, you are in Passport’s text interface where you can view alarms and enter commands.

There are three ways to log into a Passport node:

- from the local VT100 terminal or VT100 terminal emulator
- from a telnet client (on a management workstation or another Passport node)
- from the Command Console in Preside Multiservice Data Manager

When setting up your local VT100 or terminal emulator, make sure it is set to 9600 bit/s, 1 stop bit, no parity. The keyboard shortcut Control-Q may need to be pressed to resume output after connecting the local VT100 terminal or VT100 terminal emulator.

Regardless of how you log in, you must provide valid user information for that node. In some cases, the IP address of the workstation you are using must be on the valid IP address list for the node. For more information, see 241-5701-600 *Passport 7400, 15000, 20000 Configuration Guide*.

For more information on logging into a Passport node, see the following:

- “Understanding the network management interface system” (page 20)

- “Logging into a Passport node from a local operator terminal” (page 23)
- “Telnet access to a Passport node” (page 23)
- “Connecting to a device using telnet” (page 27)
- “Entering user information” (page 29)

Understanding the network management interface system

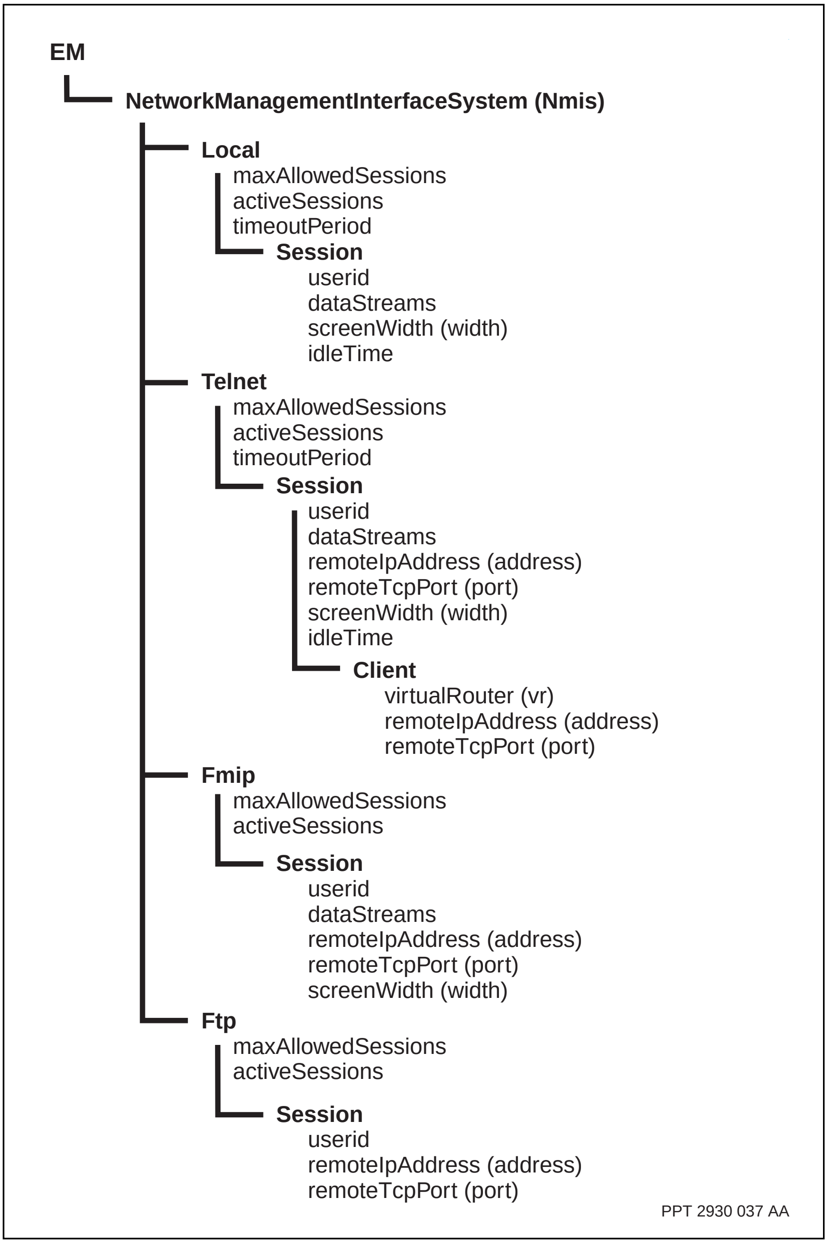
The network management interface system (NMIS) manages a number of network manager interfaces. Whenever you log into a Passport node, you use a session of a network management interface.

The figure “Network management interface components and attributes” (page 21) illustrates the components and attributes that represent the network management interface system.

See the following for more information on the NMIS:

- “Network management interfaces” (page 22)
- “Network management interface sessions” (page 22)

Figure 1
Network management interface components and attributes



Network management interfaces

There are four network management interfaces you can use to log into a Passport node:

- local (local operator)
- telnet
- FMIP
- FTP

The local operator interface allows a VT100 terminal, whether it is collocated or connected through a modem, to act as a local operator. A telnet interface allows operator sessions from a remote device through terminal emulation. An FMIP interface operates between a Passport node and Preside Multiservice Data Manager for the management of the node. The FMIP interface supports surveillance, provisioning, and operator command invocation (through the Command Console). The FTP interface lets you perform file transfers between a Passport node and a remote device. Note that both secure and non-secure FTP communication is supported. See the Preside MDM Security User Guide 241-6001-040

Each of the interfaces has a corresponding subcomponent which represents the interface. All of the subcomponents have two attributes: *maxAllowedSessions* and *activeSessions*. The *maxAllowedSessions* attribute specifies the maximum number of simultaneous sessions the interface can have. The *activeSessions* attribute specifies the number of simultaneous sessions currently active on the interface.

> ***Note:*** The local and telnet subcomponents have an additional attribute: *timeoutPeriod*. For more information, see either “Idle local operator sessions” (page 23) or “Idle telnet sessions” (page 25).

Network management interface sessions

When you log into a Passport node through a management interface, Passport dynamically creates a *Session* subcomponent to represent your connection. It automatically removes the *Session* subcomponent when you log out of the node.

Each network manager interface has the following basic session attributes:

- *userid*
- *dataStreams*
- *width*

The *userid* attribute identifies the user ID of the user logged into the session. The *dataStreams* attribute identifies which data streams (alarms, SCNs, operator logs, debug information, or rtstats) Passport displays during the operator session. The *width* attribute identifies the maximum width of the operator command responses.

Remote access sessions (telnet, FMIP, and FTP) also include information about the remote connection. The *remoteIpAddress* attribute reports the IP address of the remote device, and the *remoteTcpPort* attribute reports the TCP port number on the remote device used for the session.

Logging into a Passport node from a local operator terminal

For information on how to log into a Passport from a local-operator terminal, see 241-5701-270 *Passport 7400, 15000, 20000 Software Installation Guide*.

Idle local operator sessions

The local *timeoutPeriod* attribute is disabled by default, meaning local sessions can remain idle indefinitely. This state of inactivity uses up system resources needlessly as well as offers a security risk since unauthorized access is a possibility. All terminations due to inactivity are logged and an override capability is available. For more information about timeoutProtocol, see 241-5701-060 *Passport 7400, 15000, 20000 Components*.

Telnet access to a Passport node

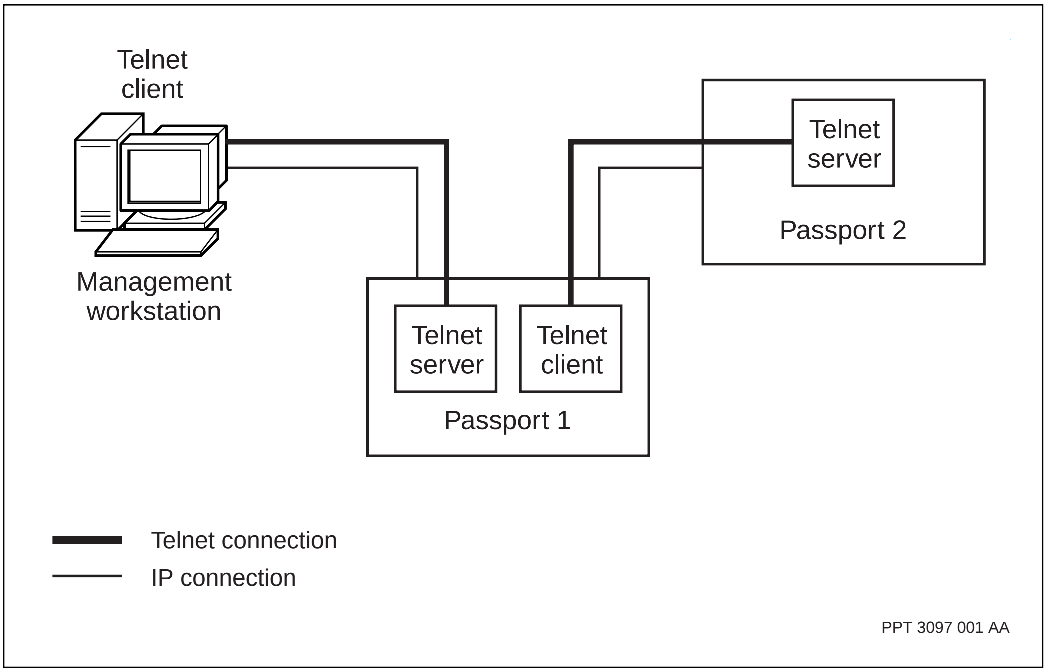
When you log into a Passport node using telnet on a management workstation, Passport acts as a telnet server. The telnet server accepts up to 12 incoming telnet connections for CP2 and up to 16 incoming telnet sessions for CP3.

> ***Note:*** CP1 is not supported.

A Passport switch can also act as a telnet client, making outgoing telnet connections to telnet servers. Since Passport can behave as both a telnet client and a server, you can use the telnet Vr command on a Passport node to connect to another Passport node. You can also use the telnet Vr command to connect to any device supporting standard telnet access. The Passport node or device must be accessible through a management or customer virtual router.

The figure “Using telnet on a Passport node” (page 24) illustrates how you can use the telnet Vr command to connect from one Passport node to another. First you telnet to Passport 1 from a management workstation using a telnet client application. Once you are logged into the telnet server on Passport 1, you can use the telnet Vr command to establish a telnet connection to Passport 2. You are now using the telnet client on Passport 1 to connect to the telnet server on Passport 2.

Figure 2
Using telnet on a Passport node



Telnet client and server sessions

Passport can act as both a telnet client and a telnet server. It acts as a server when you use a telnet application to log into a Passport node. Up to 12 different users can simultaneously log into a Passport equipped with CP2. Up to 16 different users can simultaneously log into a Passport equipped with CP3.

> ***Note:*** CP1 is not supported.

Passport acts as a telnet client when you use the telnet Vr command to establish a connection from a Passport node to another Passport node, or from a Passport node to any other device supporting standard telnet access.

An instance of a *Session* component represents a single telnet server connection. Once you have a telnet server connection, you can use the telnet Vr command to establish a telnet client connection. A *Client* component represents this outgoing connection. The attributes of the *Client* component report the IP address and TCP of the remote device, as well as the virtual router used to access the remote device.

For more information on telnet Vr, see 241-5701-050 *Passport 7400, 15000, 20000 Commands*.

Idle telnet sessions

On Passport, up to 12 telnet sessions can be simultaneously active on a Passport equipped with CP2 and up to 16 telnet sessions can be simultaneously active on a Passport equipped with CP3. Sessions that are idle continue to use system resources. As a result, the *timeoutPeriod* attribute is introduced to allow users with *systemAdministration* access to determine how long telnet sessions can remain idle before they are terminated. This attribute is disabled by default, however, it can be provisioned to be anywhere between 5 and 120 minutes. By provisioning a value for this attribute, idle sessions will be terminated after a certain length of time so that new sessions can be initiated. Additionally, when sessions are idle there is a possibility for unauthorized access. This risk is reduced with the introduction of the *timeoutPeriod* attribute.

The *timeoutPeriod* is applicable to all sessions that are created after this attribute is set. The setting of the *timeoutPeriod* is limited to users with a command impact of *systemAdministration*. If the value of the *timeoutPeriod* is changed, only new sessions created after the activation will be affected.

Each session has a unique *idleTime* value indicating the amount of time the session has been idle, even if the *timeoutPeriod* is disabled. If the *timeoutPeriod* is provisioned, a warning will automatically be issued when an idle session is within one minute of its *timeoutPeriod*.

> ***Note:*** The *idleTime* attribute begins counting idle time after the user has received a final response and has not entered any new commands. It only applies to telnet sessions in server mode. Sessions that make use of telnet client, as well as sessions that run in startup mode, are exempt from inactivity tracking.

All terminations due to inactivity are logged and an override capability is available. The override capability allows users with a command impact of *systemAdministration* to disable the *timeoutProtocol* for specific user IDs. Those user IDs for which the *timeoutProtocol* has been disabled are exempt from inactivity tracking. The *timeoutProtocol* is enabled by default; therefore, when a *timeoutPeriod* has been configured, it applies to all telnet sessions.

Connecting to a device using telnet

Connect to Passport via telnet when working from a remote location.

Prerequisites

- The node has a properly configured virtual router
- Your user ID is allowed outgoing telnet access. For information on allowing outgoing telnet access, see NN10600-605 *Passport - MDM Network Security: Operations*.
- The IP address of the device you are connecting to must be accessible through a management or customer virtual router on the node. In other words, the IP address must be within the address space of the specified virtual router.
- You must have a user ID and password for the remote device
- If you are connecting to another Passport node and that node has IP address checking enabled, the IP address of your node must be on the valid IP address list

Procedure steps

1 Connect to a Passport node or any other device supporting standard telnet access:

```
telnet -ipAddress(<remoteAddress>) Vr/<n>
```

2 When prompted, enter a user ID and password for the remote device. See “Entering user information” (page 29).

If you are connected to another Passport node, the connection is transparent. Command responses and alarms appear on screen as if you had directly connected to the node from a management workstation. If you are uncertain which Passport node you are connected to, use the me command.

Variable definitions

Variable	Value
<n>	The instance number of the virtual router that can route to the IP address.
<remoteAddress>	The IP address of the remote device to which you want to connect.

Entering user information

When you log into a Passport node, you must provide user information consisting of a valid user ID and password. Passport allows three invalid login attempts. After three invalid attempts, you must wait one minute before attempting to login again.

1 Enter your user ID at the `Enter login` prompt.

2 Enter a password at the `Enter password` prompt.

You have now logged into the Passport node.

Chapter 3
Passport provisioning system

This section covers the following topics about the Passport provisioning system:

- “Passport text interface” (page 31)
- “Provisioning views” (page 37)
- “Changing the configuration using the provisioning system” (page 55)

Passport text interface

In the Passport text interface, you can enter commands to configure the node, control the state of system, and perform diagnostic tests. The text interface has two modes: provisioning and operational. In provisioning mode, you enter commands to configure the node. In operational mode, you enter commands to control the state of the system and to perform diagnostic tests. In both modes, you can view alarms, which indicate faults.

The following sections describe the characteristics of the Passport text interface:

- “Operational mode” (page 32)
- “Provisioning mode” (page 32)
- “Command and response lines” (page 33)
- “Keyboard shortcuts” (page 35)
- “Alarms” (page 36)

Operational mode

When you initially log into a Passport node, you are in operational mode. Passport uses the following command prompt when you are in operational mode:

```
#>
```

where:
# is a sequential number assigned to commands

In operational mode, you work with operational components and attributes. In operational mode, you can

- list operational components and display operational attributes to determine the current operating parameters for the node
- control the state of parts of the node by locking and unlocking components
- set certain operational attributes and enter commands to perform diagnostic tests

For information on operational attributes, see 241-5701-060 *Passport 7400, 15000, 20000 Components*.

Provisioning mode

To change from operational mode to provisioning mode, use the start Prov command. Only one user can be in provisioning mode at a time. Passport uses the following command prompt whenever you are in provisioning mode:

```
PROV #>
```

where:
# is the current command number

In provisioning mode, you work with the provisionable components and attributes which contain the current and future configuration of the node. You can add and delete components, as well as display and set provisionable attributes. You can also verify your changes and then activate them as the new node configuration. You end provisioning mode and return to operational mode using the end Prov command.

For information on provisionable attributes, see 241-5701-060 *Passport 7400, 15000, 20000 Components*.

Command and response lines

There are four pieces of information for every command:

- command line
- component
- command response
- status line

The figure “Sample command and response” (page 34) provides an example of a sample command and all its associated information. The command line is where you enter a Passport command. The prompt for the command line changes depending on whether you are in operational mode (#>) or provisioning mode (PROV #>). In this example, the prompt indicates provisioning mode. For information on commands and their syntax, see 241-5701-050 *Passport 7400, 15000, 20000 Commands*.

After you enter a command, Passport first responds with the full name of the component affected by the command. In the example, the command affects the *Shelf Card/0* component.

Following the component name is the particular response of the command. Depending on the command, the response can be one line or pages of information. In the example, the command responds with the provisionable attributes of the *Shelf Card/0* component.

The response is followed by a status line. The status line reports the status of the command (ok or command failed) and the date and time (in the YYYY-MM-DD HH:MM:SS.SS format).

If the command is unsuccessful, the status line indicates that the command failed and the response provides details on why it failed. When a command fails because you enter it incorrectly, Passport replaces the command response with the syntax error information. Syntax error information contains the following two pieces of information:

- invalid syntax

- input

The figure “Sample syntax error” (page 34) provides a sample of an incorrectly entered command. The invalid syntax line describes the syntax error. The description has the incorrect part (for example, verb, component, or attribute) of the command in curly brackets ({}) followed by an explanation. In the example, the command contains the incorrect component name *Cord* (instead of *Card*). The invalid syntax line explains that Passport does not recognize the component name.

The input line repeats the command you entered with curly brackets ({}) around the part of the command that Passport cannot interpret. In the example, the unrecognized word Cord has curly brackets around it.

Figure 3
Sample command and response

```
  command line     PROV 2> display Shelf Card/0
component name     Shelf Card/0
      response        cardType = CPeE
                      configuredLPs = Lp/0
                      sparingConnection = notApplicable
                      commentText       = ""
   status line     ok             XXXX-09-24 15:39:54.95
```

Figure 4
Sample syntax error

```
4> lock Shelf Cord/0
Shelf
Invalid syntax: {component name} unexpected, value
unrecognized.
         Input: lock Shelf {Cord}/0
command failed XXXX-09-24 15:42:24.12
```

Keyboard shortcuts

The text interface provides a set of keyboard shortcuts you can use in either provisioning or operational mode. The shortcuts allow you to edit the command line and control the display of command responses. The table “Keyboard shortcuts” (page 35) summarizes these shortcuts.

Table 1
Keyboard shortcuts

Key	Description
Up Arrow and Down Arrow	Recalls recently used commands for edit or reuse. Passport stores the last 10 commands in a queue. Each time you press the Up Arrow key, you step back through the command queue. The Down Arrow key steps forward through the command queue.
Left Arrow and Right Arrow	Moves the cursor over the command without affecting the characters in the command.
Backspace or Delete	Deletes the character preceding the cursor.
Control-D	Deletes the character under the cursor.
Control-A	Moves the cursor to the beginning of the command line.
Control-E	Moves the cursor to the end of the command line.
Control-H	Moves the cursor back one character at a time.
Control-J or Control-M	Inserts a paragraph return.
Control-K	Deletes text from the character under the cursor to the end of the command line.
Control-N	Moves the cursor down one line at a time.
Control-P	Moves the cursor up one line at a time.
Return or Enter	Sends the command to Passport for processing.
(Sheet 1 of 2)	

Table 1 (continued)
Keyboard shortcuts

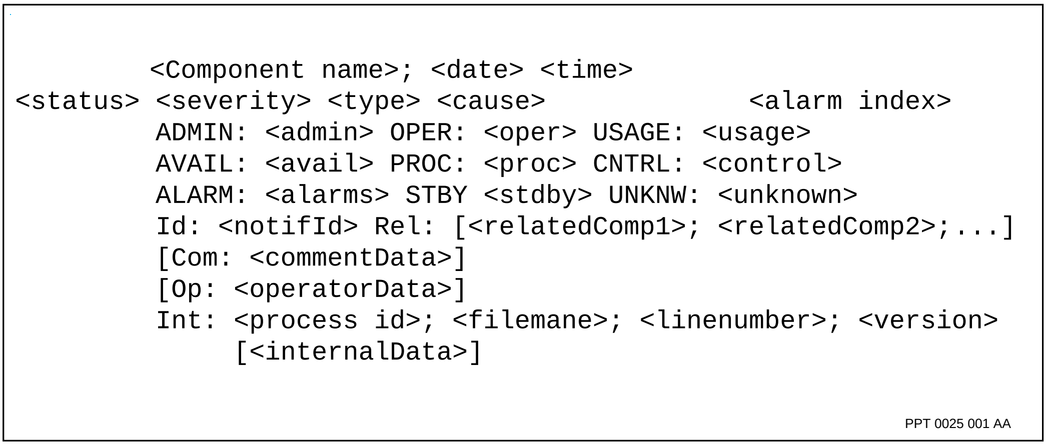
Key	Description
Control-S	Suspends the response from a command. This shortcut is useful where a command response is longer than 24 lines. If you suspend output for a long period of time, the text interface discards some subsequent output. If the session is registered to display a data stream (alarm, SCN, log, or debug data), the session discards all the data generated while response is suspended. If you suspend a telnet interface for a long time and a large amount of subsequent output is queued, the underlying TCP connection can terminate, bringing down the telnet session.
Control-Q	Resumes suspended output.
Control-C	Cancels a response. One Control-C cancels only the current response, and does not affect other queued responses.
(Sheet 2 of 2)	

Alarms

Whenever Passport detects an error or a significant event, it displays an alarm on the text interface. The alarm contains information about the component detecting the problem or event, the state of the component when it generated the alarm, and a description of the condition.

The figure “Alarm format” (page 37) illustrates the format of a Passport alarm displayed on a text interface device. For detailed information on the alarm format, see 241-5701-500 *Passport 6400, 7400, 15000, 20000 Alarms*.

Figure 5
Alarm format



```
         <Component name>; <date> <time>
<status> <severity> <type> <cause>             <alarm index>
         ADMIN: <admin> OPER: <oper> USAGE: <usage>
         AVAIL: <avail> PROC: <proc> CNTRL: <control>
         ALARM: <alarms> STBY <stdby> UNKNW: <unknown>
         Id: <notifId> Rel: [<relatedComp1>; <relatedComp2>;...]
         [Com: <commentData>]
         [Op: <operatorData>]
         Int: <process id>; <filemane>; <linenumber>; <version>
              [<internalData>]
```

PPT 0025 001 AA

Provisioning views

The Passport provisioning system lets you safely modify the configuration of your node. All the data that makes up a Passport configuration is called a view. Using the provisioning system, you can load views, save views, modify views, and activate a view as the current node configuration. The figure “Provisioning system components and attributes” (page 39) illustrates the components and attributes that represent the provisioning system.

Passport has four types of views:

- current view
- edit view
- committed view
- saved view

The current view is the configuration that is currently running on your node. The edit view is the view that you are currently modifying using the provisioning system and that can potentially become the next current view. A saved view is the view that was saved on the file system. The committed view is the view Passport uses when the node restarts.

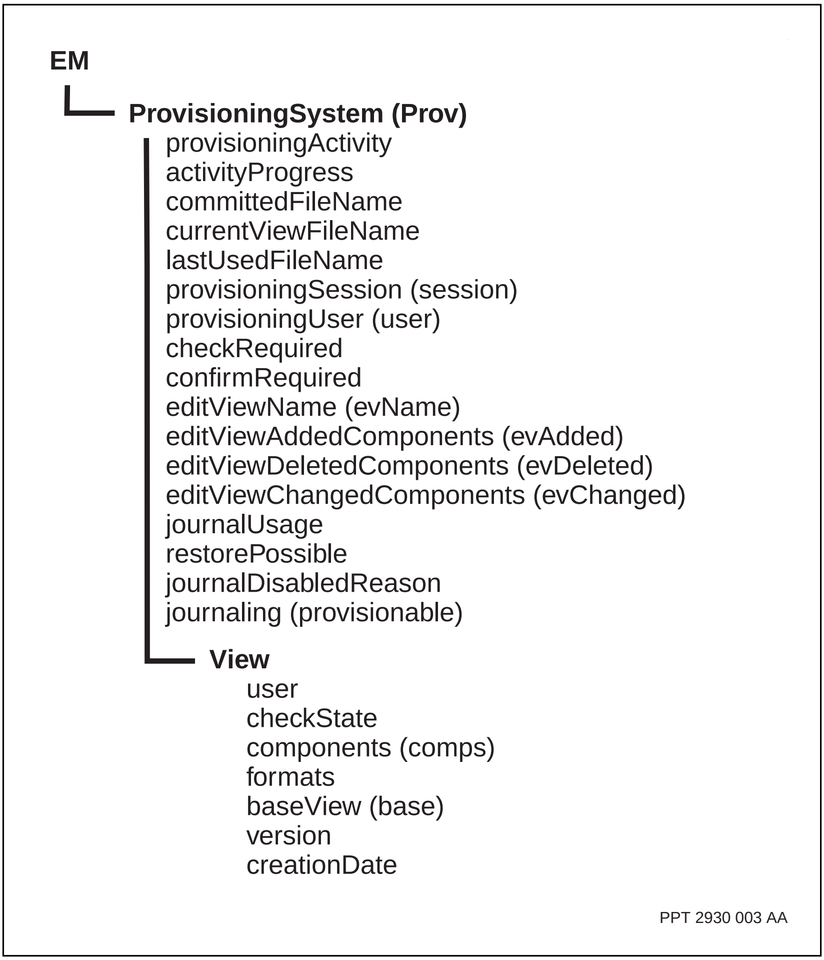
The current and edit views are stored in memory. The saved and committed views are stored on the file system. The figure “Location of Passport views” (page 40) illustrates this relationship.

For more information on the Passport provisioning system, see the following sections:

- “Current view” (page 40)
- “Edit view” (page 41)
- “Committed view” (page 43)
- “Saved view” (page 43)
- “Provisioning commands” (page 46)
- “View development example” (page 51)

For procedures on using the provisioning system, see “Changing the configuration using the provisioning system” (page 55).

Figure 6
Provisioning system components and attributes



EM

ProvisioningSystem (Prov)

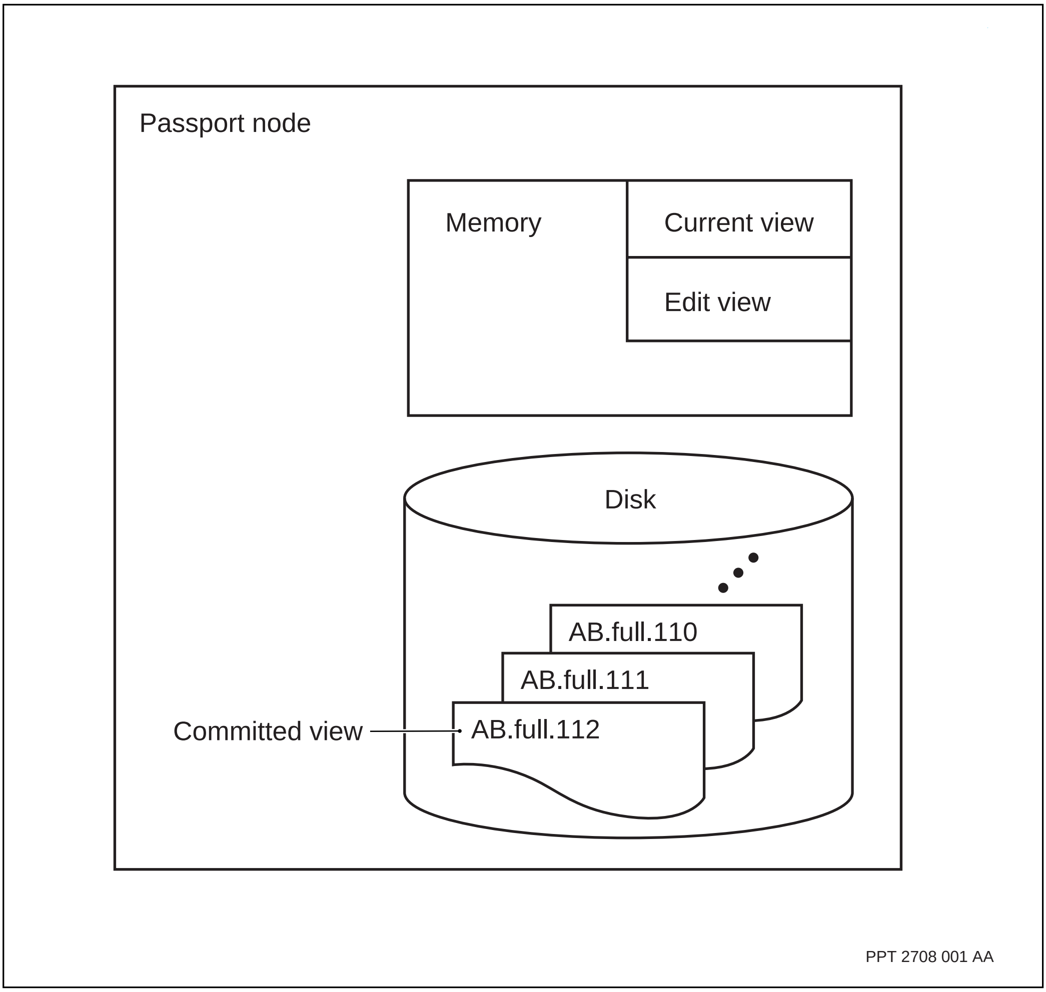
provisioningActivity
activityProgress
committedFileName
currentViewFileName
lastUsedFileName
provisioningSession (session)
provisioningUser (user)
checkRequired
confirmRequired
editViewName (evName)
editViewAddedComponents (evAdded)
editViewDeletedComponents (evDeleted)
editViewChangedComponents (evChanged)
journalUsage
restorePossible
journalDisabledReason
journaling (provisionable)

View

user
checkState
components (comps)
formats
baseView (base)
version
creationDate

PPT 2930 003 AA

Figure 7
Location of Passport views



Current view

The current view contains the set of provisioned and operational data that defines the current configuration and operating parameters of the Passport node.

The provisioning system controls the provisioned data (provisioned components and attributes) which stores the current configuration. You can view and modify the operational data of the current view (operational components and attributes), but you cannot save it directly to the file system. All operational data, including any modification you have made, is lost when the node restarts, resets, or a control processor switchover occurs.

When you make configuration changes, you do not make them directly to the current view. You modify the edit view using the provisioning system. When you are satisfied with the edit view, you activate it as the new current view. If node recovery is enabled, then the changes you made are automatically journaled to the file system. For more information about node recovery, see “Node recovery” (page 49).

You can perform the following actions on the current view using the provisioning system:

- store the current view as a saved view
- commit the current view as the view to use if the node restarts
- copy the current view into the edit view

You perform these actions using provisioning commands. For more information, see “Provisioning commands” (page 46).

The only way to change the provisionable components and attributes of the current view is to activate either the edit view or saved view. If you activate an edit or saved view but do not commit it, the committed view becomes the current view on a node reset, restart, or control processor switchover.

The *currentViewFileName* attribute of the *ProvisioningSystem* component indicates the name of the current view if it has been saved to the file system.

Edit view

The edit view is the set of provisioned data representing a node configuration which you can modify. You can make changes to the edit view using the provisioning system without affecting the current node configuration (current view). When you are satisfied with the edit view you can activate it, making it the current view.

Before you can activate an edit view, you must perform a semantic check on it. The semantic check verifies that all the settings in the edit view are self-consistent. If there are problems, the semantic check identifies them and indicates how to fix them. The semantic check also indicates any consequences of activating the edit view. For example, activating an edit view can cause a node restart.

To work with the edit view, you must start a provisioning session by entering into provisioning mode. The *provisioningUser* attribute of the *ProvisioningSystem* component indicates the user ID of the user currently in provisioning mode. Once in provisioning mode, you can perform the following actions to the edit view:

- apply changes stored in a saved view to the edit view
- delete non-permanent components
- add components
- set provisioned attribute values
- copy the current view into the edit view
- perform a semantic check of the edit view
- save the edit view to a saved view
- activate the edit view as the current view, making it the current node configuration

You perform most of these actions using provisioning commands. For information, see “Provisioning commands” (page 46).

The *editViewName* attribute of the *ProvisioningSystem* component indicates the name of the edit view if it has been saved. The *editViewAddedComponents*, *editViewDeletedComponents*, and the *editViewChangedComponents* attributes count the components in the edit view that were added, deleted, or changed from the current view.

When the node restarts, the edit view becomes identical with the current view and the commit view. Unless you save the edit view, any changes you made are lost.

Committed view

The committed view is a saved version of a current view that the node uses when it resets, restarts, or switches over from the active to the standby control processor.

The committed view provides a configuration that is known to work and will allow the node to initialize, as well as connect to the network. This configuration is important when you are trying to activate new configurations. If the new configuration is faulty, the committed view provides a configuration to which the node can return (roll back). After activating a new configuration, you have 20 minutes to confirm that the new configuration has successfully initialized and that the node is operating properly. If you do not confirm the activation within 20 minutes, Passport automatically restarts the node using the committed view.

In some instances, a committed view is not identical to the current view. When you activate an edit view as the current view, the current view and the committed view are different. The current view contains newly updated configuration data, while the committed view contains configuration data based on a previous current view.

Only when you commit the current view are the two views identical. Before you can commit the current view, you must save it to the file system. If you have a dual-CP node, the disks must also be synchronized.

The *committedFileName* attribute of the *ProvisioningSystem* component indicates the file name of the committed view.

Saved view

A saved view is a copy of the current view, edit view, or committed view stored on the file system.

You can save an edit view so you can reopen and change it later. Saving the edit view also means that your configuration changes are not lost in the event of a node restart.

An instance of the *View* component (a subcomponent of *ProvisioningSystem*) represents each saved view stored on the file system. The attributes of the *View* component detail the saved view, including its type, formats, and semantic check state.

There are two types of saved views:

- full saved views
- partial saved views

A full saved view includes all configuration data from either the edit or current view. You can load it into the edit view and activate it as the current view. You can also directly activate a full saved view as long as you semantically checked it before saving it.

A partial saved view includes all the configuration data for a component and all its subcomponents from either the edit or current view. You can load the component from a partial saved view into the edit view to add the component or change an existing component's configuration. You cannot activate a partial saved view.

A single saved view can have several formats. The format of the saved view depends on the options you used when saving the view and whether or not the view is the committed view. The table "Format of saved views" (page 45) describes the possible formats for a saved view.

Saved views are stored in subdirectories and files in the /provisioning directory. When you save a view, Passport assigns it a name, which becomes a subdirectory of the provisioning directory. The subdirectory contains the view in its various formats. For information on naming saved views, see 241-5701-050 *Passport 7400, 15000, 20000 Commands*.

Table 2
Format of saved views

Saved view format	Associated view	Contents and characteristics	Storage format
portable	Current or edit views	Includes all configuration data. A portable saved view can be moved from one Passport node to another, where you can load and further modify it.	Passport internal
ASCII	Current or edit views	Includes all configuration data. You can use views in ASCII format with non-Passport tools to generate configuration printouts.	ASCII
commit	Committed view	Includes all configuration data in the committed view. This format is in a Passport-internal format that enables fast activation.	Passport internal
delta	Edit view	Includes the changes made between the current view and the edit view. This format loads and saves faster than the other formats. You can also apply the changes stored in a delta view to the edit view. You can activate a view stored in this format only if activating it does not require a system restart.	Passport internal
part (partial)	Current or edit views	Includes all information about a component, including all its subcomponents and associated attributes. You can use views in partial format with non-Passport tools to generate configuration printouts.	ASCII
(Sheet 1 of 2)			

Table 2 (continued)
Format of saved views

Saved view format	Associated view	Contents and characteristics	Storage format
pre-4.X provisioning files	Current and edit view	Includes all information about the Passport configuration. You can convert Pre-4.X provisioning files to portable or ASCII format (depending on the view type) by loading them into the edit view on the current release.	Passport internal
Note: Pre-4.X files are used with the Passport 7400 series only.			
(Sheet 2 of 2)			

Provisioning commands

You work with the provisioning system through a set of provisioning commands. These commands allow you to load and save views, modify and activate the edit view, as well as confirm and commit the current view.

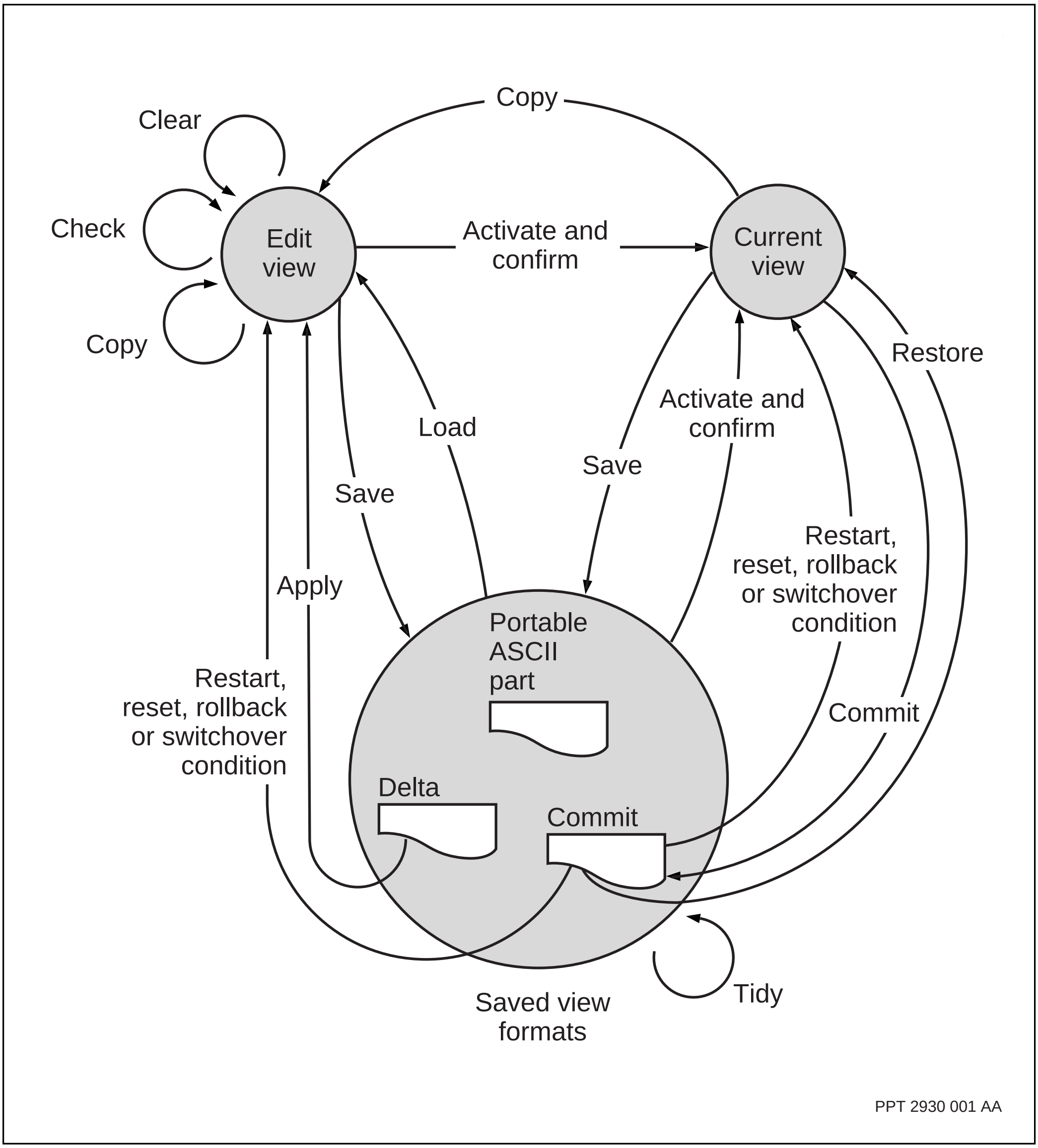
The provisioning system commands are

- activate Prov
- apply Prov
- check Prov
- clear Prov
- commit Prov
- confirm Prov
- continue Prov
- copy Prov
- end Prov
- load Prov
- restore Prov
- save Prov
- start Prov

- stop Prov
- tidy Prov

The figure “Relationship between views and provisioning commands” (page 48) illustrates the relationship between provisioning commands and Passport views.

Figure 8
Relationship between views and provisioning commands



Node recovery

The node recovery feature introduces automatic journaling of the current view. That is, a journal log file is created during each manual activation and contains a delta of the configuration changes between the current and edit views. This enables the current view to be manually recovered in the case of a switch reset through the restore Prov command, even if the current view has not been explicitly saved by the user. This feature, in conjunction with the MDM Node Recovery tool, also enables the current view to be efficiently backed-up and restored in the case of a complete switch failure.

Only configuration changes that have been confirmed can be restored. Note that journal log files are not saved for null activations (where no configuration changes have been made) or for activations that will result in a switch reload, such as a software migration.

The saving of journal log files can become disabled in the following cases:

- when journal log file saving is disabled via provisioning
- when node recovery is enabled after being disabled, an initial commit must be done before the provisioning system can save journal log files (since journals are based on the committed view)
- too many log files were created since last the commit Prov was performed
- a file system error occurs (for example, disk is corrupted or disk is full)
- the file system is locked
- when it is the first manual activation after a switch reload and the current view is different than the committed view (for example, after a software migration, but before the commit)
- when the activation results in a switch reload activation (for example, a software migration activation)

When a new provisioning view is committed, all pre-existing journal log files are automatically removed by the system since the current view is now equal to the committed view and is thus saved.

The journal log files are saved in the following directory: /provisioning/journal/current. Note that this directory only exists if there are journal log files are created.

For information on how to enable or disable node recovery, see "Disabling and enabling saving of journal log files" (page 50).

For information about Node recovery on Preside Multiservice Data Manager, see, 241-6001-807 *Preside MDM Passport/SNMP Devices Backup and Restore User Guide*.

The *maxNumberJournalFiles* attribute of the *ProvisioningSystem* component specifies the maximum number of journal files that can be saved.

The *currentJournal* attribute of the *ProvisioningSystem* component indicates the number of journal log files that have been saved since the last Commit Prov command was issued.

The *restorePossible* attribute of the *ProvisioningSystem* component indicates if it is possible to perform the restore Prov is possible. For more information about this command, see 241-5701-050 *Passport 7400, 15000, 20000 Commands*.

The *journalDisabledReason* attribute of the *ProvisioningSystem* component indicates whether or not journaling is disabled.

Disabling and enabling saving of journal log files

By default, the saving of journal log files is enabled. By disabling journal log files, the saving of journal log files will be turned off for any subsequent activations. By enabling journal log files, the saving of journal log files will be turned on for any subsequent activations.

> ***Note:*** You need to commit the configuration after enabling or disabling journal log file saving.

1 Enter provisioning mode:

```
start Prov
```

2 If you want to enable journaling:

```
set Prov maxNumberJournalFiles <enabled>
```

where:
<enabled> is a number in the range of 100 to 2000.

3 If you want to disable journaling:

```
set Prov maxNumberJournalFiles none
```

4 Verify the provisioning changes you have made:

```
check Prov
```

5 Activate the edit view:

```
activate Prov
```

6 Confirm that the activation was successful:

```
confirm prov
```

The confirm command verifies that the newly activated edit view allows proper access to the node. If you do not confirm the activation within 20 minutes, the node automatically restarts using the committed view.

7 If you want the new configuration (which is now in the current view) to become the permanent configuration for the node, commit it:

```
commit -file <filename> Prov
```

If you do not commit the current view, the next time the node restarts it uses the previously committed view, and the changes you just activated are lost.

8 End the provisioning session:

```
end Prov
```

View development example

This example illustrates how the provisioning system uses current, edit, committed, and saved views during a provisioning session.

Your Passport node starts using the committed saved view today.full.001. At this point, the current view and edit view are identical. You display the names of the current and edit view using the following command:

```
display Prov currentViewFileName, editViewName
```

You receive the following response:

```
Prov
   currentViewFileName = today.full.001
   editViewName        = today.full.001
```

You display the formats of the today.full.001 saved view using the following command:

```
display Prov View/today.full.001 formats
```

You receive the following response:

```
Prov View/today.full.001
   formats = commit port
```

Because today.full.001 is the committed view, it uses the commit format. The portable (port) format means that today.full.001 is not based on any other saved view. A committed view cannot be based on another view, so it must always be saved in the portable format.

You start provisioning mode and make changes to the edit view. You then save the edit view using the following command:

```
save Prov
```

You receive the following response:

```
Prov
   Saving the edit view into today.full.002 (with delta
formats)...
   The provisioning data is saved in file
today.full.002.
```

The edit view is now stored in a new saved view (today.full.002) using the delta format. Since today.full.001 was the current view when you saved, today.full.002 is based on today.full.001. You verify the format and the base view of today.full.002 using the following command:

```
display Prov View/today.full.002 formats, baseView
```

You receive the following response:

```
Prov View/today.full.002
   formats = delta
   baseView = today.full.001
```

You now make more changes to the edit view and save it again. It is now stored in today.full.003 using the delta format. Since today.full.001 is still the current view, it is also the base view for today.full.003. You verify the format and base view of today.full.003 using the following command:

```
display Prov View/today.full.003 formats, baseView
```

You receive the following response:

```
Prov View/today.full.003
   formats = delta
   baseView = today.full.001
```

You decide to save the edit view in ASCII format so you can use it with a non-Passport tool. You enter the following command:

```
save -ascii Prov
```

You receive the following response:

```
Prov
   Saving the edit view into today.full.003 (with ascii
formats)...
   The provisioning data is saved in file
today.full.003.
```

Since you did not change the edit view, the provisioning system does not create a new saved view. It just adds the ASCII format to the today.full.003 saved view. You verify the formats using the following command:

```
display Prov View/today.full.003 formats
```

You receive the following response:

```
Prov View/today.full.003
   formats = ascii delta
```

You now go through the process of checking, activating, and confirming the edit view (which is stored in today.full.003). The edit view becomes the current view. You commit the current view using the following command:

```
commit Prov
```

You receive the following response:

```
Prov
   Saving the current view into today.full.004 (with
commit, portable formats)
   ...
Prov
   The committed file is today.full.004.
```

Since the committed view cannot be based on another view, the commit Prov command automatically saves the current view using the portable format. The command creates a new saved view (today.full.004) to record the checked status of the edit view. It then stores today.full.004 using the commit format. You verify the formats, check state, and base view of today.full.004 using the following command:

```
display Prov View/today.full.004 formats, checkState,
baseView
```

You receive the following response:

```
Prov View/today.full.004
   formats = commit port
   checkState = full
   baseView = none
```

Displaying saved views on the disk

Before you load a saved view into the edit view, you can check the names and characteristics of all saved views stored on the disk.

1 Display the saved views currently on disk:

```
display Prov View/*
```

This command displays, in a table format, each saved view currently on the disk along with the user who created it. The following fields are also displayed in the table:

- check state (failed, unknown, partial, softwareChanged, or full)
- format (ascii, commit, delta, part, or portable)
- the associated software version (the version of base software running when the view was saved)

2 Display all the characteristics of a particular saved view:

```
display Prov View/<view_name>
```

where:
<view_name> is the name of a saved view.

Changing the configuration using the provisioning system

You must use the provisioning system to make changes to the configuration of your Passport. The provisioning system maintains an edit view, which represents a potential new configuration. After you have made changes to the edit view, you can activate it so that it becomes the current configuration, which is called the current view.

In most cases, you do not want to activate configuration changes immediately after you enter them in the edit view. Some changes require your node to restart, which causes service outages. A new configuration can also create errors when you activate it. For these reasons, enter a number of configuration changes and then activate all of them during a scheduled service period.

The following sections explain how to enter and activate your configuration changes:

- “Entering configuration changes” (page 56)
- “Activating and committing configuration changes” (page 60)

If you need to make immediate provisioning changes to isolate or correct network problems, see “Configuring for immediate activation” (page 64). To check the names and characteristics of the saved views stored on disk, see “Displaying saved views on the disk” (page 54).

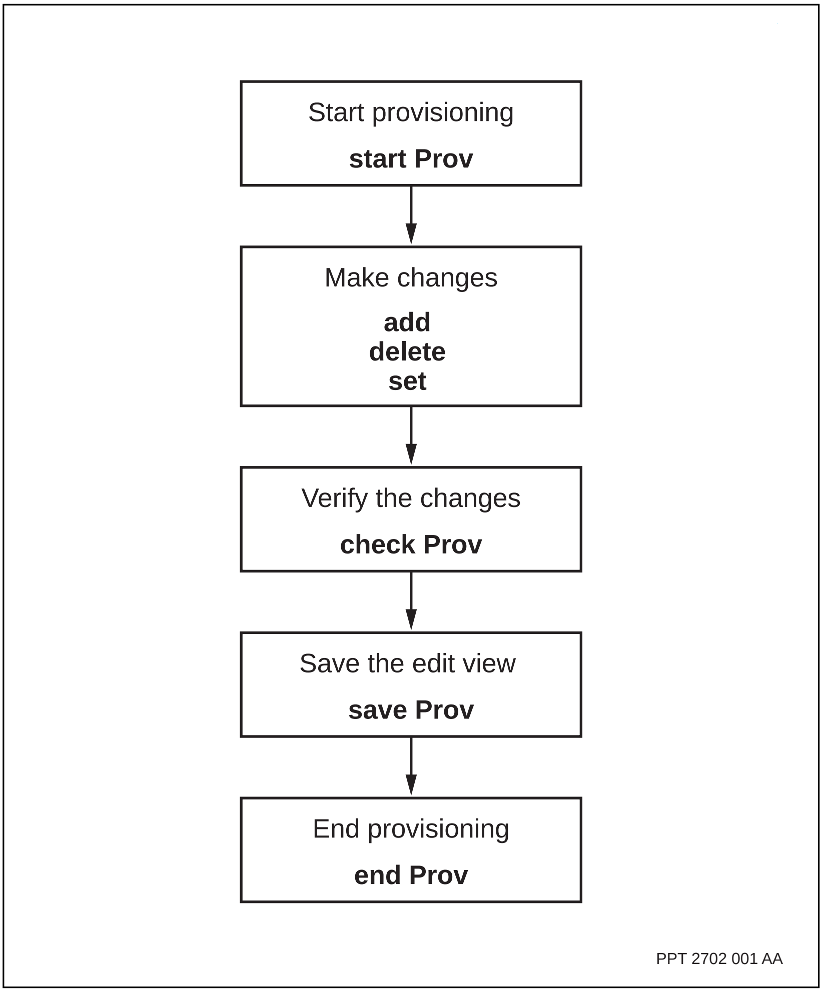
Entering configuration changes

Only one user at a time can make configuration changes. Passport stores any changes you make to the edit view for the next user making configuration changes. To ensure that your changes are not lost, save the edit view before ending your provisioning session.

Entering configuration changes flow chart

The figure “Flowchart for entering configuration changes” (page 57) illustrates the steps to follow when entering configuration changes.

Figure 9
Flowchart for entering configuration changes



Procedure steps

1 Enter provisioning mode:

```
start Prov
```

When you enter provisioning mode, Passport indicates the differences, if any, between the current and the edit view. Passport reports added and deleted components as well as changed attributes.

2 If you want to discard any previous changes made to the edit view, copy the current view into the edit view:

```
copy Prov
```

The edit view and the current view are now identical.

3 Make changes to the edit view by adding and deleting components, and setting attribute values:

```
add <component_name>
delete <component_name>
set <component_name> <attribute> <value>
```

4 Verify the provisioning changes you have made:

```
check Prov
```

Correct any errors before continuing, and then verify the provisioning changes again.

You can speed up the verification by checking only components in the edit view that are different from those in the current view.

```
check -changed Prov
```

Note: This command does not check the components that are dependent upon the changed components.

5 Save the edit view:

```
save Prov
```

Note: This step is optional if journaling is enabled because journaling automatically saves the configuration data. For example, if journaling is enabled, your configuration data would not be lost when a switch reset occurs.

6 End the provisioning session:

```
end Prov
```

To activate the provisioning changes you have just entered, see “Activating and committing configuration changes” (page 60).

Variable definitions

Variable	Value
<attribute>	The name of the attribute
<component_name>	The name of the component, including its instance value.
<value>	The value you are assigning the attribute

Activating and committing configuration changes

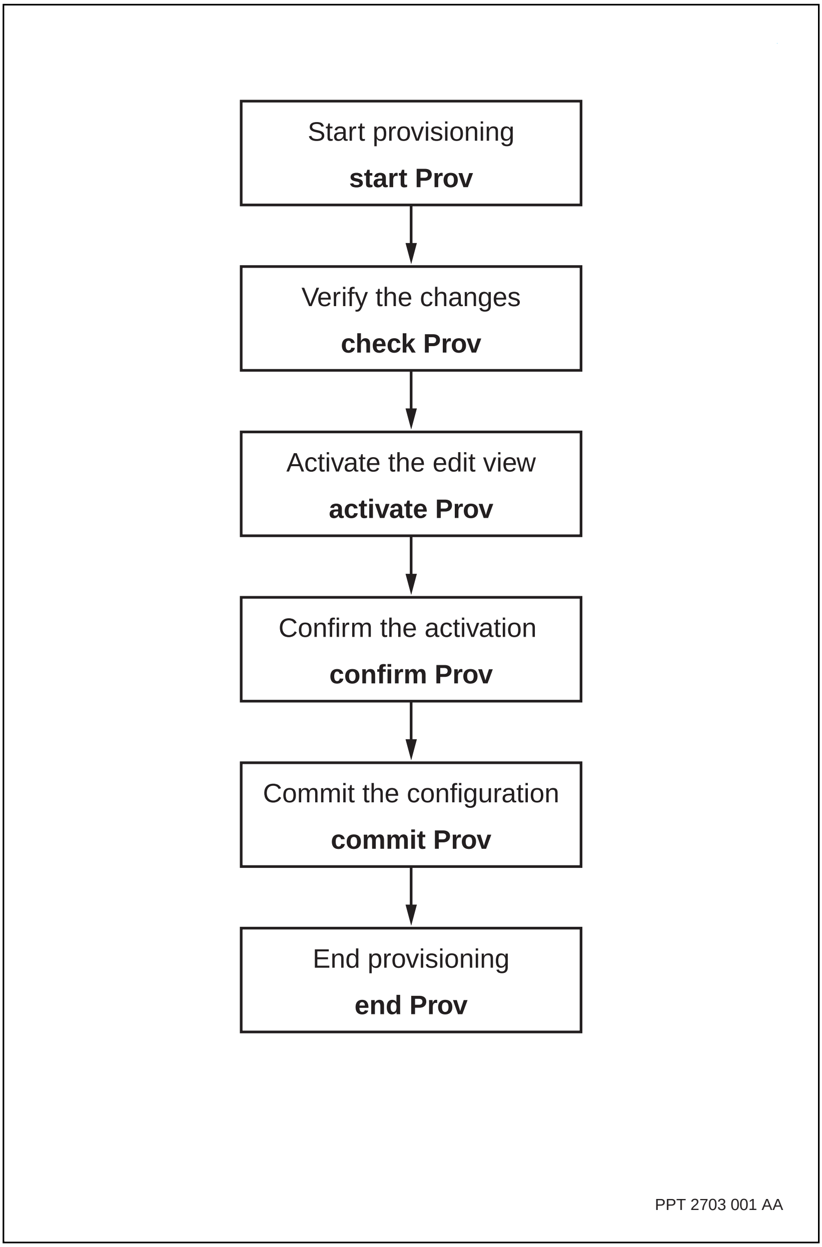
Configuration changes become effective only after you activate them. During activation, the edit view becomes the current view and any changes made to the edit view become active. After you activate the new configuration from the edit view, you must confirm that the activation was successful.

Once you have confirmed the activation, you can make the new configuration the committed view. The committed view is the permanent configuration of the node. When a Passport node restarts, it uses the configuration in the committed view. If you do not commit the view, the changes will be in effect only for the current configuration and will be lost on a subsequent restart, reset, or CP switchover.

Activating and committing configuration change flow chart

The figure “Flowchart for activating configuration changes” (page 61) illustrates the steps you follow when activating configuration changes.

Figure 10
Flowchart for activating configuration changes



Procedure steps

1 Enter provisioning mode:

```
start Prov
```

When you enter provisioning mode, Passport indicates the differences, if any, between the current and the edit view. Passport reports added and deleted components, as well as changed attributes.

2 If the edit view does not contain the configuration changes you want to activate, load the configuration changes from a saved view:

```
load -file(<view_name>) Prov
```

3 Verify the configuration changes:

```
check Prov
```

Correct any errors before continuing, and then verify the provisioning changes again.

Passport warns you if activating these configuration changes requires a service, function processor, or the entire node to restart. If there is a restart, service outage and potential data loss results.

4 Optionally, save the edit view:

```
save Prov
```

5 Activate the edit view:

```
activate Prov
```

When the activation is complete, the current view and the edit view are identical.

Note: If activating the edit view causes the node to restart, wait until the node comes back up, then log on and enter provisioning mode again (using the start Prov command).

6 Confirm that the activation was successful:

```
confirm prov
```

The confirm command verifies that the newly activated edit view allows proper access to the node. If you do not confirm the activation within 20 minutes, the node automatically restarts using the committed view.

7 If you want the new configuration (which is now in the current view) to become the permanent configuration for the node, commit it:

```
commit Prov
```

If you do not commit the current view, the next time the node restarts, resets, or CP switchover occurs, it uses the previously committed view, and the changes you just activated are lost.

Note: If the changes you have activated include changes to the software running on the node, you will have to verify, activate, and confirm the changes again (see step 3 to step 6).

8 End the provisioning session:

```
end Prov
```

Variable definitions

Variable	Value
<view_name>	The name of the saved view. To determine which saved views are available, see “Displaying saved views on the disk” (page 54).

Configuring for immediate activation

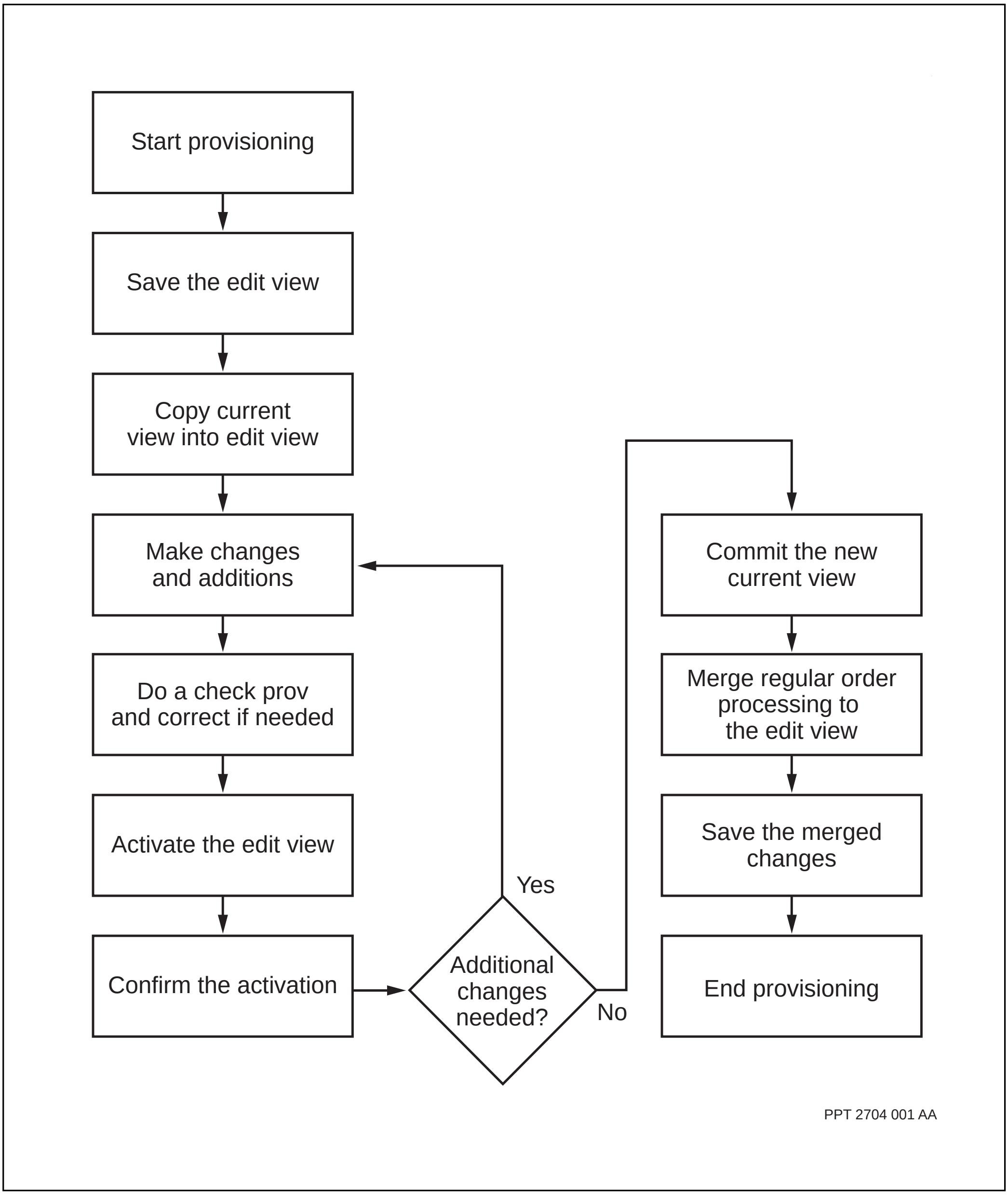
Sometimes when you are troubleshooting, you need to make immediate configuration changes to isolate or correct a network problem. In this situation, you activate only specific configuration changes, not the accumulated changes stored in the edit view.

When you undertake configuration for immediate activation, save all the accumulated changes in the edit view so you can make specific configuration changes.

Configuring for immediate activation flow chart

Once you have completed your immediate activation, apply the saved changes back into the edit view. The figure “Flowchart for immediate configuration” (page 65) illustrates the process.

Figure 11
Flowchart for immediate configuration


Start provisioning
Save the edit view
Copy current view into edit view
Make changes and additions
Do a check prov and correct if needed
Activate the edit view
Confirm the activation
Additional changes needed?
Yes
No
Commit the new current view
Merge regular order processing to the edit view
Save the merged changes
End provisioning
PPT 2704 001 AA

Procedure steps

1 Enter provisioning mode:

```
start Prov
```

2 Store the edit view as a saved view:

```
save Prov
```

Record the filename of the saved view. The filename appears when the command completes.

3 Discard all previous changes made to the edit view and copy the current view into the edit view:

```
copy Prov
```

The edit view and the current view are now identical.

4 Make changes to the edit view by adding and deleting components, and setting attribute values:

```
add <component_name>
delete <component_name>
set <component_name> <attribute> <value>
```

5 Verify the configuration changes:

```
check Prov
```

Correct any errors before continuing, and then verify the configuration changes again.

Passport warns you if activating these configuration changes requires a service, function processor, or the entire node to restart. If there is a restart, service outage and potential data loss results.

6 If you want to create a new file containing these changes, save the edit view:

```
save Prov
```

7 Activate the edit view:

```
activate Prov
```

When the activation is complete, the current view and the edit view are identical.

Note: If activating the edit view causes the node to restart, wait until the node comes back up, then log on and enter provisioning mode again (using the start Prov command).

8 Confirm that the activation was successful:

```
confirm Prov
```

The confirm command verifies that the newly activated edit view allows proper access to the node. If you do not confirm the activation within 20 minutes, the node will automatically restart using the committed view.

9 If you want the new configuration (which is now in the current view) to become the permanent configuration for the node, commit it:

```
commit Prov
```

If you do not commit the current view, the next time the node restarts it uses the previously committed view and the changes you just activated are lost.

Note: If the changes you have activated include changes to the software running on the node, you have to verify, activate, and confirm the changes again (see step 5 to step 8).

10 Apply the changes you saved at step 2 to the edit view:

```
apply -file(<view_name>) Prov
```

Watch the messages that appear when you apply the saved view. The messages indicate any conflicts between the changes you have just made and those stored in the saved view. Resolve these conflicts before continuing.

11 Verify the configuration changes:

```
check Prov
```

Correct any errors before continuing, then verify the configuration changes again.

12 Save the edit view:

```
save Prov
```

13 End the provisioning session:

```
end Prov
```

Variable definitions

Variable	Value
<attribute>	The name of the attribute
<component_name>	The name of the component, including its instance value.
<value>	The value you are assigning the attribute
<view_name>	The name of the view you saved at step 2

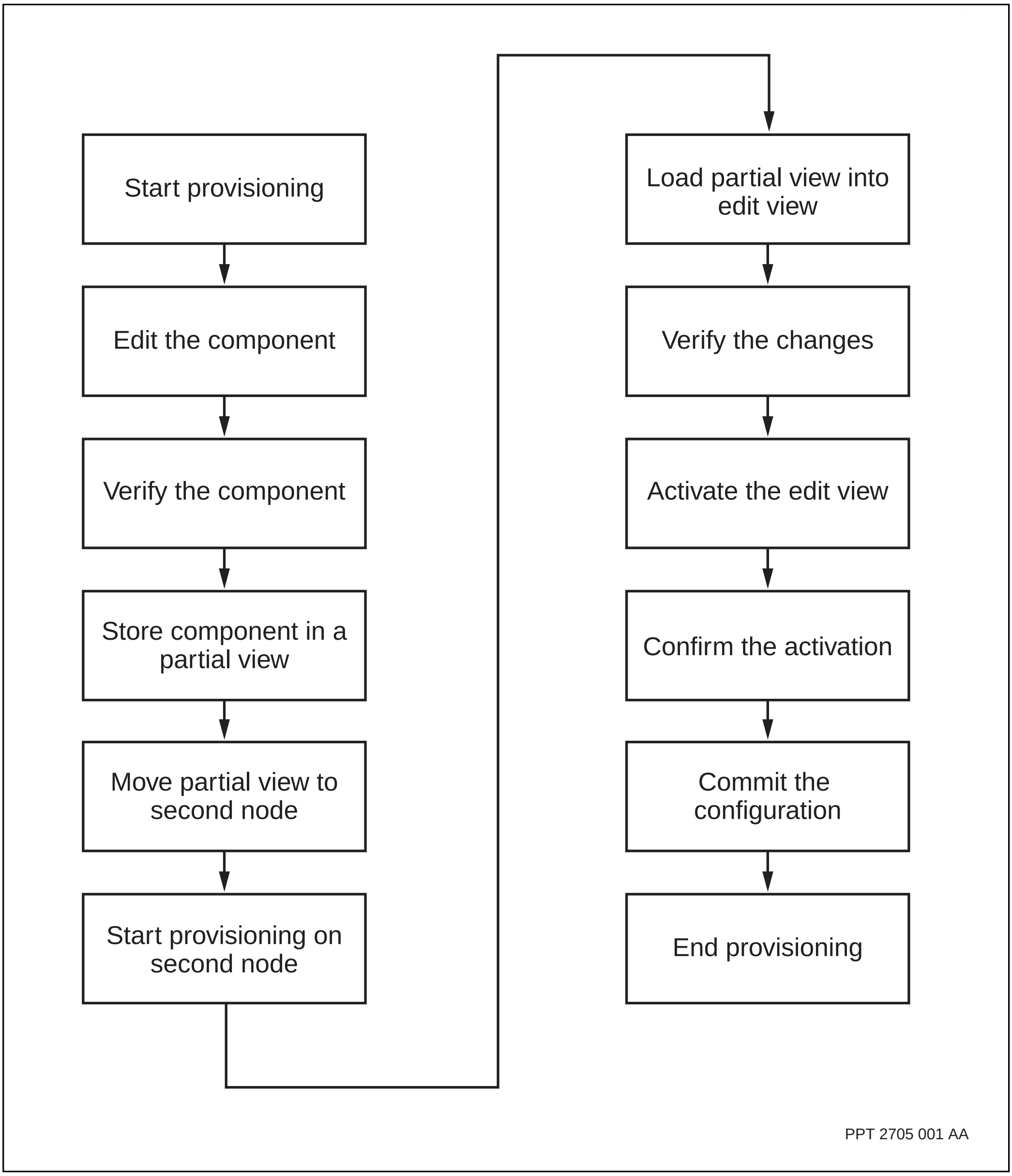
Copying a component configuration to another node

Partial saved views can be used for developing a configuration for a component on one node and then using that configuration on other nodes in the network.

Copying a component configuration to another node flow chart

The figure “Flowchart for using partial views” (page 70) illustrates the steps for developing a configuration for a component on one node and then using that configuration on other nodes in the network.

Figure 12
Flowchart for using partial views



Procedure steps

1 Start provisioning mode:

```
start Prov
```

2 Make changes to a component:

```
add <component_name>
set <component_name> <attribute> <value>
```

3 Verify the changes you have made to the component:

```
check -component(<component_name>) Prov
```

Correct any errors before continuing, and then verify the component again.

4 Save the component in a partial saved view:

```
save -file(<filename>) -component(<component_name>)
Prov
```

5 Transfer the partial saved view to a second node using FTP. For a description of the FTP process, see 241-6001-023 *Preside MDM Configuration Management for Passport User Guide*.

Note: You must move the view's entire subdirectory.

6 Start provisioning on the second node:

```
start Prov
```

7 Load the partial view into the edit view on the second node:

```
load -file(<filename>) Prov
```

8 Verify the configuration changes:

```
check Prov
```

Correct any errors before continuing, and then verify the configuration changes again.

Passport warns you if activating these configuration changes requires a service, function processor, or the entire node to restart. If there is a restart, service outage and potential data loss results.

9 Activate the edit view:

```
activate Prov
```

When the activation is complete, the current view and the edit view are identical.

Note: If activating the edit view causes the node to restart, wait until the node comes back up, then log on and enter provisioning mode again (using the start Prov command).

10 Confirm that the activation was successful:

```
confirm Prov
```

The confirm command verifies that the newly activated edit view allows proper access to the node. If you do not confirm the activation within 20 minutes, the node automatically restarts using the committed view.

11 If you want the new configuration (which is now in the current view) to become the permanent configuration for the node, commit it:

```
commit Prov
```

If you do not commit the current view, the next time the node restarts it uses the previously committed view and the changes you just activated are lost.

12 End the provisioning session:

```
end Prov
```

Variable definitions

Variable	**Value**
<attribute>	The name of the attribute
<component_name>	The name of the component, including its instance value.
<filename>	The base software name of the saved view.
<value>	The value you are assigning the attribute

Passport 7400, 15000, 20000

Management System User Interface Guide

Release 5.2

Publication: 241-5701-045
Document status: Standard
Document version: 5.2S1
Document date: November 2003
Printed in Canada