VitalQIP® DNS/DHCP & IP Management Software

VitalQIP® (QIP) | Release 7.2

Web Service Interface Specification

190-409-091R7.2
July 2009
Issue 3

Trademarks

All trademarks and service marks specified herein are owned by their respective companies.

Licenses

Refer to Appendix C, “Third party software license statements” in the *VitalQIP Release 7.2 Installation Guide* (190-409-043R7.2) for a complete description of all software licenses used to develop this product.

Contents

List of tables

About this document

Purpose

This document is intended for anyone who needs to use the VitalQIP Web Service.

Refer to this preface for the audience, organization, and typographical conventions used in this manual. The preface also describes the package contents, how to order additional manuals, and how to obtain technical support.

Reason for revision

The following table shows the revision history of this document.

Issue	Revision	Location
3	Support for UDA definition and UDA groups added.	• Table 2-1, "Action operations and messages" (p. 2-4) • "UDA definition" (p. 2-240) • "UDA group definition" (p. 2-248) • "UDA_DEFINITION_ATTRIBUTE_TYPE" (p. 3-37) • "UDA_DEFINITION_INFRASTRUCTURE_TYPE" (p. 3-38) • "UDAGROUP" (p. 3-39) • "UDALIST" (p. 3-40) • "VQIPManager_AssociateListRequest operation" (p. 4-18) • "VQIPManager_DissociateListRequest operation" (p. 4-29) • "VQIPManager_ GetAssociatedComponentsRequest operation" (p. 4-40) • "VQIPManager_SequenceAssociatedComponentsRequest operation" (p. 4-59) • "VQIPManager_SetUdaRequest operation" (p. 4-61) • "UdaDefinition.xsd" (p. 6-106)

Issue	Revision	Location
3	Support for Access Control added.	• "Access Control Devices" (p. 2-36) • "Access Control Subscribers" (p. 2-40) • "AC_SUBSCRIBER_STATUS_TYPE" (p. 3-3) • "AcDevice.xsd" (p. 6-4) • "AcSubscriber.xsd" (p. 6-6)
3	Schema example added for Managed Files.	• "ManagedFile.xsd" (p. 6-77)
3	Support for Clone Server added.	• "Clone server" (p. 2-146) • "Request for Clone DNS Server" (p. 4-21)
3	Clarified support for operations supported on IPv4 subnets.	• "IPv4 Subnet (Get and Search only)" (p. 2-3)
2	Support for UsernameToken Profile added	• "Client – VitalQIP Web Service interaction" (p. 1-4) • "Incoming/outgoing message structure format" (p. 2-6) • "Sample Request and Response messages" (p. 4-1)
2	Support for AfriNIC added.	• "AfriNIC reporting" (p. 2-123) • "REGISTRY_TYPE" (p. 3-23) • "RIR_REPORT_TYPE" (p. 3-27) • "Block.xsd" (p. 6-11)

Intended audience

This manual is intended for personnel who are responsible for implementing a southbound SOAP interface to the VitalQIP web client.

How to use this document

This manual is organized as follows:

Chapter 1: Introduction	This chapter contains an introduction to the VitalQIP Web Service.
Chapter 2: Web Service operations	This chapter provides information on the operations supported by the VitalQIP Web Service application.
Chapter 3: Enumeration types	This chapter contains information on the data structures used by the VitalQIP Web Service.
Chapter 4: Sample Request and Response messages	This chapter provides sample request and response messages for each supported operation.
Chapter 5: Error messages	This chapter lists the various error messages that may be received by the northbound system.

Chapter 6: VitalQIP Web Service schema	This chapter describes the VitalQIP Web Service schema.
Chapter 7: VitalQIP Web Service WSDL	This chapter contains the VitalQIP Web Service WSDL.

Conventions used

This guide uses the following typographical conventions:

Appearance	Description
italicized text	• File and directory names • Emphasized information • Titles of publications • A value that the user supplies
graphical user interface text or key name	• Text that is displayed in a graphical user interface or in a hardware label • The name of a key on the keyboard
`input text`	Command names and text that the user types or selects as input to a system
`output text`	Text that a system displays or prints

Related information

The following documents describe the VitalQIP product:

- *VitalQIP Administrator Reference Manual* (part number: 190-409-042R7.2)

  This guide describes planning and configuring your network, information about the VitalQIP interface, advanced DNS and DHCP configurations, and troubleshooting.
- *VitalQIP Installation Guide* (part number: 190-409-043R7.2)

  This guide describes how to install the VitalQIP product.
- *VitalQIP Command Line Interface User's Guide* (part number: 190-409-044R7.2)

  This guide discusses and describes how to use the VitalQIP Command Line Interface.
- *VitalQIP User's Guide* (part number: 190-409-068R7.2)

  This guide describes how to set up and use the VitalQIP user interface.
- *VitalQIP Web Client User's Guide* (part number: 190-409-079R7.2)

  This guide describes how to use the VitalQIP web client.

Product Training Support

Alcatel-Lucent University offers cost-effective educational programs that support the VitalQIP product. Our offerings also include courses on the underlying technology for the VitalQIP products (for example, DNS and DHCP). Our classes blend presentation, discussion, and hands-on exercises to reinforce learning. Students acquire in-depth knowledge and gain expertise by practicing with our products in a controlled, instructor-facilitated setting. If you have any questions, please contact us at 1-888-LUCENT8, Option 2, Option 2.

Technical support

If you need assistance with VitalQIP, you can contact the Technical Assistance Center for your region. Contact information is provided in the following table.

Region	Address	Contact information
North America	Alcatel-Lucent 400 Lapp Road. Suite 101 Malvern, PA 19355 USA	Phone: 1-866-LUCENT8 (582-3688) Option 1, Option 2 Web: **https://support.lucent.com**
Europe, Middle East, and Africa	Alcatel-Lucent Voyager Place Shoppenhangers Road Maidenhead Berkshire SL6 2PJ UK	Phone: 00 800 00 LUCENT or +353 1 692 4579 E-mail: *emeacallcenter@alcatel-lucent.com* Web: **https://support.lucent.com**
Central and South America	Alcatel-Lucent Calle 10, No. 145 San Pedro de los Pinos, 01180 Ciudad de Mexico Mexico	Phone: Mexico 01 800 123 8705 or (52) 55 5278 7235 Brazil 0800 89 19325 or (55) 193707 7900 Argentina 0800 666 1687 Venezuela 0 800 1004136 Costa Rica 0800-012-2222 or 1800 58 58877 For other local CALA numbers, consult the web site or contact your local sales representative. Web: **https://support.lucent.com**

Region	Address	Contact information
Asia Pacific	Alcatel-Lucent Australia 280 Botany Road Alexandria NSW 2015 Australia	Phone: 1800-458-236 (toll free from within Australia) (IDD) 800-5823-6888 (toll free from Asia Pacific - China, Hong Kong, Indonesia, South Korea, Malaysia, New Zealand, Philippines, Singapore, Taiwan, and Thailand) (613) 9614-8530 (toll call from any country) E-mail: *apactss@alcatel-lucent.com*

How to order

To order Alcatel-Lucent documents, contact your local sales representative or use the Online Customer Support Site (OLCS) web site (http://support.lucent.com).

How to comment

To comment on this document, go to the Online Comment Form (http://infodoc.alcatel-lucent.com/comments/) or e-mail your comments to the Comments Hotline (comments@alcatel-lucent.com).

1 Introduction

Overview

Purpose

This chapter contains an introduction to the VitalQIP Web Service.

Contents

This chapter covers these topics.

VitalQIP Web Service

Overview

This document describes the VitalQIP Web Service that is used to access the Address Allocation, Address Management , DNS Views (including their zones and servers), and Managed Files. functionality provided by VitalQIP. The interface uses the Simple Object Access Protocol (SOAP), which encodes messages using Extensible Markup Language (XML), and uses HTTP or HTTPS for transport. This document contains the Web Services Description Language (WSDL) document that defines the interface. The WSDL document specifies interface operations, their associated messages and message parameters, the schema for the data types required by the messages, port types, and binding information. Descriptions of this WSDL information are presented in this document, along with examples.

Purpose

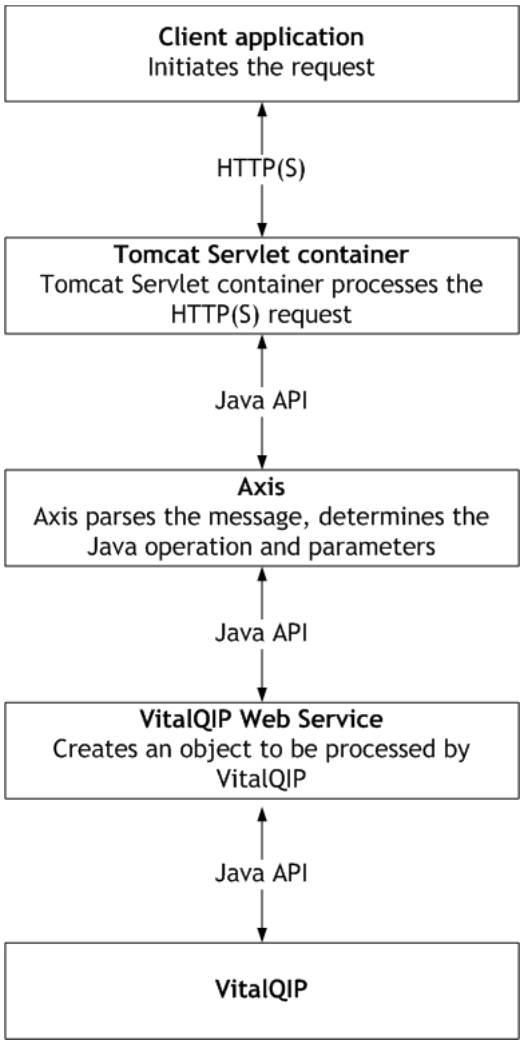
This document is intended for anyone who needs to use the VitalQIP Web Service. In order to gain a high level understanding of the interface, refer to the following paragraph and Chapter 4, “Sample Request and Response messages”. Developers from the northbound system will find the entire document to be relevant.

Mechanism used in interface

The VitalQIP Northbound Interface is implemented as a Web Service and is therefore referred to as VitalQIP Web Services Application in the rest of the document. The term Web Service describes a standardized way of integrating Web-based applications using the XML, SOAP, WSDL over an Internet protocol backbone. XML is used to tag the data, SOAP is used to transfer the data, WSDL is used for describing the services available. Web Services allow organizations to communicate data without in-depth knowledge of each other's IT systems even if the systems are behind a firewall. SOAP is an XML-based communication protocol and encoding format for inter-application communication. Originally conceived by Microsoft and Userland software, it has evolved through several generations. The current specification is version 1.2 though version 1.1 is more widespread. SOAP is widely viewed as the backbone of Web Services. There are various implementations of SOAP such as Apache Axis, Apache Axis2, gSOAP, TclSOAP, to name a few. VitalQIP uses Apache Axis2 version 1.3.1. Apache Axis2 is a Java based implementation that supports SOAP 1.1 and 1.2 specifications. Implemented services in the VitalQIP Web Services application can be deployed in the Apache Axis server. The deployed services enable Axis to recognize, process and invoke requested service operations on the back-end service implementation. Tomcat servlet engine executes Axis as a web application and provides the necessary HTTP/HTTPS messaging framework. VitalQIP Web Service uses a multi-threaded Java process. Methods implemented by the VitalQIP Web Service are invoked by Axis.

The following illustration shows the steps involved in the VitalQIP Web Service Application receiving a client request.

Figure 1-1 Steps involved in receiving a client request



While Axis is integrated in the existing Tomcat servlet container being used by VitalQIP, it is independent of existing servlets in the container, such as the servlet that processes GUI requests.

As indicated in Figure 1-1, a Web Service is described by a WSDL document. A WSDL binding describes how the service is bound to the SOAP messaging protocol. A WSDL SOAP binding can be either a RPC style binding or a document style binding. A SOAP binding can also have an encoded use or a literal use. Of those possible choices, the VitalQIP Web Service supports WS-I Basic Profile compliant Document/Literal style.

The WSDL for the interface to be used by the VitalQIP Web Service is included in Chapter 7, "VitalQIP Web Service WSDL".

Client – VitalQIP Web Service interaction

In the following subsections, client refers to the client application, that is, the northbound system that interacts with the VitalQIP Web Service application.

Authentication

Each request requires authentication information such as valid username, password and organization of VitalQIP administrator with appropriate permission. This information should be passed in as input to each operation. The request is rejected if either username or password is not valid for the specified organization or if the user does not have permissions for the requested action.

Web Services Security (WSSE) defines a standard way of encoding username and password in a security header in SOAP messages. This is known as Username Token. As of VitalQIP 7.2, the username and password information, formerly in CommonInfo, has been relocated into Username Token. Refer to "Header" (p. 2-7) for details.

Multiple Concurrent Connections

Multiple clients can connect concurrently with the VitalQIP Web Service Application. Requests for each connection are processed independently, so there is no guaranteed ordering between requests received from different connections. However, requests from one particular client connection are handled in first-in-first-out order. Maximum number of concurrent connections allowed is configurable.

Logging

The VitalQIP Web Service uses Log4J software for logging data that can be used to debug and to get information on operations being performed by the service. Log4J supports various logging levels, such as WARN, DEBUG, and so on.

Interoperability with clients

The client must be designed to be compliant with SOAP version 1.1 or 1.2 (**http://www.w3.org/TR/SOAP/**), WSDL version 2.0 (**http://www.w3.org/TR/wsdl**), and Web Services Security UsernameToken Profile 1.0 (**http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0.pdf**).

SOAP clients already using the existing VitalQIP 7.0 or 7.1 Web Service are strongly recommended to update to VitalQIP 7.2 WSDL. Alternatively, you can continue to run as-is after enabling the transitional 7.0 or 7.1 Web Service only provided in 7.2. Refer to "To enable compatibility with VitalQIP 7.0/7.1 Web Service" in the *VitalQIP 7.2 Administrator Reference Manual* to enable VitalQIP 7.0 or 7.1 Web Service. If operating in transitional mode, then continue to use Web Service Specification for VitalQIP 7.0 or 7.1.

URL

VitalQIP 7.2 Web Service

VitalQIP 7.2 Web Service based API can be accessed using the following URL:

`http://<webServerAddr>:<webServerPort>/ws/services/VQIPWebService`

- ***<webServerAddr>*** is either the hostname or IP address of the VitalQIP web server.
- ***<webServerPort>*** is the TCP/IP port on which the VitalQIP web server is configured to listen. The default value is 80.

  Note: If the VitalQIP web server is configured to use SSL, the URL must start with **https:** and not **http:**.

VitalQIP 7.0/7.1 Web Service

The transitional VitalQIP 7.0/7.1 Web Service continues to be accessible using the original URL:

`http://<webServerAddr>:<webServerPort>/services/VQIPManagerPort`

2 Web Service operations

Overview

Purpose

This chapter provides information on the operations supported by the VitalQIP Web Service application.

Contents

This chapter covers these topics.

Operations in VQIPManagerPortType

Overview

Purpose

This section provides information on the operations supported by the VitalQIP Web Service Application. An operation can include one or more messages. For the VitalQIP Web Service Application, each operation can have a Request message, a Response message, and a Fault message.

Each of the operations, their messages and the parameters they take are described in the following subsections. Parameters for a message can be required (indicated as ‘R’ in the tables) or optional (indicated as ‘O’ in the tables).

VitalQIP Web Service supports the following:

- Access Control Device
- Access Control Subscriber
- Address Management Blocks
- Address Pools
- Address Ranges
- Blocks
- DNS Views (Views, Zones, Servers and Access Control Lists)
- Internet Registries
- Maintainers
- Managed Files
- Nodes
- Rules
- IPv4 Subnet (Get and Search only)
- IPv6 Subnet
- Templates
- User Defined Attributes (UDA)
- User Defined Attribute Groups

Actions that are available for the above objects, along with the operation name and incoming and outgoing messages are given in the following table. Please note that for certain objects, additional actions are supported, such as Copy for Rules, and Merge for Pools. Please refer to the individual object sections for details on the supported operations.

Table 2-1 Action operations and messages

Action	Operation	Incoming message	Outgoing message
Add	VQIPManager_AddRequest	AddRequest	Response
AddContained Object	VQIPManager_AddContained Object	AddContainedObjectRequest	Response
Associate	VQIPManager_AssociateRequest	AssociateRequest	Response
AssociateList	VQIPManager_AssociateList Request	AssociateListRequest	Response
CopyRequest	VQIPManager_CopyRequest	CopyRequest	Response
CountAssociations	VQIPManager_CountAssociations	CountAssociationsRequest	CountAssociations Response
Delete	VQIPManager_DeleteRequest	DeleteRequest	Response
DeleteContained Object	VQIPManager_DeleteContained Object	DeleteContainedObject Request	Response
DeleteForced Request	VQIPManager_DeleteForced Request	DeleteForcedRequest	Response
Dissociate	VQIPManager_DissociateRequest	DissociateRequest	Response
DissociateList	VQIPManager_DissociateList Request	DissociateListRequest	Response
FindAssociations	VQIPManager_FindAssociations Request	FindAssociationsRequest	FindAssociations Response
Get	VQIPManager_GetRequest	GetRequest	GetResponse
GetAll	VQIPManager_GetAllRequest	GetAllRequest	GetAllResponse
GetAssociated Components Request	VQIPManager_GetAssociated ComponentsRequest	GetAssociatedComponents Request	GetAssociated Components Response
GetFile	VQIPManager_GetFileRequest	GetFileRequest	GetResponse
Merge	VQIPManager_MergeRequest	MergeRequest	Response
Paginated Search	VQIPManager_PageSearchRequest	PageSearchRequest	PageSearch Response
Search	VQIPManager_SearchRequest	SearchRequest	SearchResponse
Sequence	VQIPManager_SequenceRequest	SequenceRequest	Response

Action	Operation	Incoming message	Outgoing message
Sequence Associated Components Request	VQIPManager_Sequence AssociatedComponentsRequest	SequenceAssociated ComponentsRequest	Response
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

Incoming/outgoing message structure format

Overview

All incoming SOAP messages are comprised of a header and a body. As of VitalQIP 7.2, the header contains security section, with user authentication information encoded as the UsernameToken. The body contains the application request. All outgoing SOAP messages contain body only. The following illustrates the SOAP message layout.

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope
     xmlns:soapenv=http://schemas.xmlsoap.org/soap/envelope/
     xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <soapenv:Header>
    <wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-
   200401-wss-wssecurity-secext-1.0.xsd"http://...
       soapenv:mustUnderstand="1">
      <wsse:UsernameToken xmlns:wsu="http://docs.oasis-
   open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-
   1.0.xsdhttp://...">
        <wsse:Username>qipman</wsse:Username>
        <wsse:Password
Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-
   token-profile-1.0#PasswordText">qipman</wsse:Password>
        </wsse:UsernameToken>
      </wsse:Security>
      <wsa:To>http://www.example.com:8080/ws/services/VQIPWebService
      </wsa:To>
      <wsa:MessageID>urn:uuid:E614CC1FFF321E8054122</wsa:MessageID>
      <wsa:Action>VQIPManager_GetRequest</wsa:Action>
    </soapenv:Header>
    <soapenv:Body>
      <ns1:GetRequest
xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:GetRequest">
        <ns1:commonParam>
          <ns1:organization>My Org</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
          </ns1:commonParam>
          <ns1:reqObject xsi:type="ns1:RULE_REC">
            <ns1:name>abc</ns1:name>
          </ns1:reqObject>
        </ns1:GetRequest>
      </soapenv:Body>
   </soapenv:Envelope>
```

Header

The Header section of all incoming messages includes UsernameToken, identifying the requestor by username and a password for authentication. Depending on VitalQIP configuration choices, the password can be in clear text or digest form. When using clear text, consider configuring VitalQIP with SSL for greater security. The VitalQIP Web Service accepts either format. When a custom Authentication Callout is deployed in VitalQIP, the password can only be in clear text.

The details of creating UsernameToken headers depend on the SOAP toolkit chosen by the client application. Generally a WS Security plug-in is needed by the SOAP toolkit.

Body

The Body section of all incoming messages is made up of three or more parts (elements). The first part is of CommonInfo type, it contains information needed by all operations, such as organization and user locale. The second optional part is of **`JOB_REQUEST`** type; it contains job scheduling information for operations supporting scheduling. The remaining part(s) of the message are operation specific and vary from operation to operation.

CommonInfo

This structure is required for all operations. It includes parameters that are common to all operations, as described in the following table.

Table 2-2 CommonInfo parameters

Part name	Type	Opt/Req	Default	Description
`organization`	**`xsd:string`**	R	None	***Required if*** **`orgID`** ***is not populated.*** Name of the organization to which the entity being operated on belongs.
`orgID`	**`xsd:long`**	R	None	***Required if*** **`organization`** ***is not populated.*** Identifier of the organization to which the entity being operated on belongs. **Note:** Either **`organization`** or **`orgId`** must be populated.
`locale`	**`xsd:string`**	O	None	Locale for displaying error messages. If the locale is not specified then default locale of VitalQIP system is assumed.
`site`	**`xsd:string`**	O	None	Name of the site to which the entity being operated on belongs.

Request structure - JOB_REQUEST message

Overview

This structure includes parameters needed to schedule the requested job. This structure is applicable only to certain operations. Refer to "Scheduling operations" (p. 2-35) for details on which operations can be scheduled.

Table 2-3 Job_Request details

Part Name	Type	Opt / Req	Default	Description
when	**xsd:string**	O	None	Time when the job will be started. The format of the date can be any of the following formats: • MM/dd/yyyy hh:mm • yyyy-MM-dd hh:mm • yyyy-MM-dd hh:mm z • yyyyMMdd'T'hhmmZ • yyyy-MM-dd h:mma • MMM d HH:mm yyyy • EEE MMM d HH:mm:ss zzz yyyy
toEmail	**xsd:string**	O	None	Indicates the email address(es) to send email to when the job completes, comma separated.
ccEmail	**xsd:string**	O	None	Indicates the email address(es) to cc the job email to when the job completes, comma separated.
sendEmail	**xsd:boolean**	O	None	Indicates that the job should send email. If the toEmail argument is not specified, by default the email will go to the email address specified in the administrator profile. If no address is specified in the administrator profile, an error will be displayed.
runMode	**qipws:SCHEDULE_JOB_TYPE**	O	None	Determines when the request will be executed. Valid values are: • IN_FOREGROUND • IN_BACKGROUND • SCHEDULED_AT
description	**xsd:string**	O	None	Optional description of the job.

Request structure - AddContainedObjectRequest message

Overview

The following common **`AddContainedObjectRequest`** structure will be used for adding an object contained within another object. For example, for adding an interface to a node. The northbound system will use the object being added to populate the containedObject field. As the name indicates, **`parentKey`** will be used to uniquely identify the parent object to which the object identified by containedObject is being added. For example, if the northbound system wants to add an ipaddress to an interface, an object of type **`IPADDR_REC`** will be used to populate containedObject field and **`INTERFACE_KEY`** will be used to populate the **`parentKey`** field.

Table 2-4 AddContainedObjectRequest details

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, “CommonInfo parameters” (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	N/A in this release
`containedObject`	**`qipws:VQIP_BASE_ENTITY`**	R	None	Object being added.
`parentKey`	**`qipws:VQIP_KEY_ENTITY`**	R	None	Key of the parent object.

Request structure - AddRequest message

Overview

The following request structure will be used for Add operation. The northbound system will use the appropriate data type to populate the reqObject field. For example, if the northbound system wants to create a Rule, an object of type **RULE_REC** will be used to populate **reqObject** field.

Table 2-5 AddRequest details

Part Name	Type	Opt / Req	Default	Description
commonInfo	**qipws:CommonInfo**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **CommonInfo** structure.
jobReq	**qipws:JOB_REQUEST**	O	None	Applicable only to certain operations. Refer to Scheduling Operations section below for details on which operations can be scheduled.
reqObject	**qipws:VQIP_BASE_ENTITY**	R	None	Object to add.

Request structure - AssociateRequest message

The following common **`AssociateRequest`** structure will be used for associating an object to another object. Details of the request are given below:

Table 2-6 AssociateRequest details

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	N/A in this release
`srcKey`	**`qipws:VQIP_KEY_ENTITY`**	R	None	key of source object. For example, while associating a rule with a template, **`srcKey`** will be of type **`RULE_KEYS`**.
`targetKey`	**`qipws:VQIP_KEY_ENTITY`**	R	None	key of target object. For example, while associating a rule with a template, targetKey will be of type **`ADDR_TEMPL_KEYS`** or **`SUBNET_PROF_TEMPL_KEYS`**.
targetParameters	qipws:VQIP_BASE_ENTITY	O	None	Additional parameters that are only used by some targets.

Request structure - AssociateListRequest Message

Overview

The following structure is for associating a list of objects to another object. This is a specialized message only supported by a few types of targetKey. (It is not a generalization of "Request Structure - AssociateListRequest Message".) Request details follow:

Table 2-7 AssociateListRequest details

Part Name	Type	Opt/ Req	Default	Description
commonInfo	**qipws:CommonInfo**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **CommonInfo** structure.
srcKeys	**qipws:VQIP_KEY_ ENTITY**	R	None	Keys of source objects. For example, to associate a set of UDA definitions to an infrastructure, **srcKeys** will be an array of **UDA_DEFINITION_KEY**.
targetKey	**qipws:VQIP_KEY_EN TITY**	R	None	Key of the target object, for example, **UDA_INFRASTRUCTURE_TYPE _KEY**.

Request structure - CopyRequest message

Overview

The following common **`CopyRequest`** structure is used for copying an object:

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	N/A in this release
`from`	**`qipws:VQIP_KEY_ENTITY`**	R	None	Keys of source object
`to`	**`qipws:VQIP_KEY_ENTITY`**	R	None	Keys of target object

Request structure - CountAssociations message

Overview

The following common **`CountAssociationsRequest`** structure will be used for determining the number of objects associated with a given object:

Table 2-8 CountAssociationsRequest details

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	N/A in this release
`objectKey`	**`qipws:VQIP_KEY_ENTITY`**	R	None	key of the object.

Request structure - DeleteContainedObjectRequest message

Overview

The following common **DeleteContainedObjectRequest** structure will be used for deleting an object contained within another object. For example, for deleting an interface from a node. The northbound system will use the object being deleted to populate the **containedObject** field. As the name indicates, **objectKey** will be used to uniquely identify the object being deleted. For example, if the northbound system wants to delete an interface, **INTERFACE_KEY** will be used to populate the **objectKey** field.

Table 2-9 DeleteContainedObjectRequest details

Part Name	Type	Opt / Req	Default	Description
commonInfo	**qipws:CommonInfo**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **CommonInfo** structure.
jobReq	**qipws:JOB_REQUEST**	O	None	Not available in this release.
objectKey	**qipws:VQIP_KEY_ ENTITY**	R	None	Key for identifying the object being deleted.
deleteRelated Records	**xsd: boolean**	O		Flag to indicate if related records should be deleted. For example, if ipaddress is being deleted, if associated dns records should be deleted.

Request structure - DeleteRequest message

Overview

The following request structure will be used for **`Delete`** operation. The northbound system will use the appropriate data type to populate the **`reqObject`** field. For example, if the northbound system wants to delete a Rule, an object of type **`RULE_REC`** will be used to populate **`reqObject`** field.

Table 2-10 DeleteRequest details

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	N/A in this release
`reqObj`	**`qipws:VQIP_BASE_ENTITY`**	R	None	Object to delete.

Request structure - DeleteRequestForced message

Overview

The following request structure provides a variation of **`Delete`** that is only supported by a few operations. Compared to **`DeleteRequest`**, a **`DeleteRequestForced`** message uses a **`VQIP_KEY_ENTITY`** to select the record and provides a boolean to force the delete, even if the object being deleted is still in use. A northbound system will use the appropriate data type to populate the objectKey field. For example, if the northbound system wants to delete a DNS ACL Template, an object of type **`DNS_ACL_TEMPLATE_KEY`** will be populated in the **`objectKey`** field.

Table 2-11 DeleteRequestForced details

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	Not available in this release.
`objectKey`	**`qipws:VQIP_KEY_ENTITY`**	R	None	Object to delete.
`forced`	**`xsd:boolean`**	R	None	Force the delete, even if object is still in use.

Request structure - DissociateRequest message

Overview

The following common **`DissociateRequest`** structure will be used for dissociating an object from another object. Details of the request are given below:

Table 2-12 DissociateRequest details

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	Not available in this release.
`srcKey`	**`qipws:VQIP_KEY_ENTITY`**	R	None	Key of source object. For example, while associating a rule with a template, srcKey will be of type **`RULE_KEYS`**.
`targetKey`	**`qipws:VQIP_KEY_ENTITY`**	R	None	Key of target object. For example, while dissociating a rule from a template, targetKey will be of type **`ADDR_TEMPL_KEYS`** or **`SUBNET_PROF_TEMPL_KEYS`**.

Request structure - DissociateListRequestMessage

Overview

The following structure is for dissociating a list of objects from another object. (This is a specialized message only supported by a few types of targetKey.) It is not a generalization of "Request Structure - DissociateRequest Message". Request details follow:

Table 2-13 DissociateListRequest details

Part Name	Type	Opt / Req	Default	Description
commonInfo	**qipws:CommonInfo**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **CommonInfo** structure.
jobReq	**qipws:JOB_REQUEST**	O	None	Not available in this release.
srcKey	**qipws:VQIP_KEY_ ENTITY**	R	None	Keys of source objects. For example, to associate a set of UDA definitions to an infrastructure, **srcKeys** will be an array of **UDA_DEFINITION_KEY**.
targetKey	**qipws:VQIP_KEY_ ENTITY**	R	None	Key of the target object. For the above example, it is **UDA_INFRASTRUCTURE_TYPE_ KEY**.

Request structure - FindAssociationsRequest message

Overview

The following common **`FindAssociationsRequest`** structure will be used for retrieving the objects associated with a given object:

Table 2-14 FindAssociations details

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	Not available in this release.
`objectKey`	**`qipws:VQIP_KEY_ENTITY`**	R	None	Key of the object for which to retrieve the associations.

Request structure - GetAllRequest message

Overview

The common request structure described in the following table is used for **`GetAll`** transactions. As the name suggests, the **`GetAll`** transaction retrieves all records for a particular VitalQIP object type. The object types for which this transaction is supported are:

- DHCP Option templates
- DHCP Scope Policy templates
- DHCP Servers
- Domains
- DNS Servers
- DNS Views
- Organizations
- Node types
- User Classes
- Vendor Classes

The northbound system uses the **`VQIPManager_GetAll`** operation with **`VQIP_GetAllRequestMessage`**. Details of **`VQIP_GetAllRequestMessage`** are given in the following table.

Table 2-15 VQIP_GetAllRequestMessage details

Part name	Type	Opt/Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	Not available in this release.
`listType`	**`qipws:LIST_TYPE`**	R	None	Refer to "LIST_TYPE" (p. 3-18) for definition of **`LIST_TYPE`**.

Request structure - GetAssociatedComponentsRequest

Overview

The following request structure retrieves a list of components associated with the infrastructure type. This is a specialized call that only has meaning for a few infrastructure types. For example, use it to obtain the list of UDA and UDA groups associated with the specified infrastructure type.

Table 2-16 GetAssociatedComponents details

Part name	Type	Opt/Req	Default	Description
commonInfo	**qipws:CommonInfo**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **CommonInfo** structure.
param	**qipws: VQIP_KEY_ENTITY**	R	None	Key identifying infrastructure type.

Request structure - GetFileRequest message

Overview

The following request structure is used to request file data from the server. A northbound system will use the appropriate data type to populate the reqObject. For example, to request a managed file, use a **`MANAGED_FILE_KEY`**.

Managed files are expected to contain printable data. As such, the end of line character differs between UNIX and Microsoft clients; UNIX uses a single \n (linefeed) character, whereas Windows uses the two characters \r\n (carriage return+linefeed). Set the OS type as needed.

Table 2-17 GetFileRequest details

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	N/A in this release
`reqObject`	**`qipws:VQIP_KEY_ENTITY`**	R	None	An object to retrieve, reference by a key.
`os`	**`qipws:OS_TYPE`**	R	None	UNIX or WINDOWS. Controls end of line handling.

Request structure - GetRequest message

Overview

The following request structure will be used to retrieve attributes of an object from VitalQIP. The northbound system will use the appropriate data type to populate the **`reqObject`** field. For example, if the northbound system wants to retrieve a Rule, an object of type **`RULE_REC`** will be used to populate **`reqObject`** field.

Table 2-18 GetRequest details

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	Not available in this release.
`reqObject`	**`qipws:VQIP_BASE_ENTITY`**	R	None	An object to retrieve.

Request structure - MergeRequest message

Overview

The following common **`MergeRequest`** structure should be used to merge two objects. The merge operation is supported for merging pools, address blocks and IPv6 subnets only.

Table 2-19 MergeRequest details

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	Not available in this release.
`from`	**`qipws:VQIP_BASE_ENTIRY`**	R	None	Structure identifying the source object (pool, address block, v6 subnet) being merged.
`to`	**`qipws:VQIP_BASE_ENTITY`**	R	None	Structure identifying the destination object, i.e. the object being merged into. Both source and destination structures must be of same type.

Request structure - MoveObjectRequest message

Overview

The following common **`MoveObjectRequest`** structure should be used to move an object such as an interface or an ipaddress to a different object.

Table 2-20 MoveObjectRequest details

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	Not available in this release.
`objKey`	**`qipws:VQIP_KEY_ENTITY`**	R	None	Key for identifying the object being moved
`toKey`	**`qipws:VQIP_KEY_ENTITY`**	R		Key for identifying the destination object.

Request structure - PageSearchRequest message

Overview

The common request structure described in the following table is used for Search transactions that return responses in a paginated format. The northbound system uses the appropriate data type to populate the **reqObject** field. For example, if the northbound system wants to invoke a Search operation for blocks, an object of type **SEARCH_ADDRESS_BLOCK_REC** is used to populate the **reqObject** field. Number of records per page can be specified, along with an offset into the set of total pages. For example, if the northbound system wishes to include 20 records per page and the access record # 30 in one operation, the **pageSize** should be set to 20 and **pageCount** should be set to 2.

Table 2-21 PageSearchRequest structure

Part name	Type	Opt/Req	Default	Description
commonInfo	**qipws:CommonInfo**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **CommonInfo** structure.
jobReq	**qipws:JOB_REQUEST**	O	None	Not available in this releas.e
reqObject	qipws:VQIP_BASE_ ENTITY	O	None	Object being searched for and its search criteria.
pageSize	**xsd:int**	O	200	Number of records per page.
pageCount	**xsd:int**	O	1	Offset into the set of total pages.

Request structure - SearchRequest message

Overview

The following common request structure is used for search operation for templates, maintainers and rules objects which return results in non-paginated list. The northbound system will use the appropriate data type to populate the **`reqObject`** field. For example, if the northbound system wants to invoke search operation for Rules, an object of type **`RULE_REC`** will be used to populate **`reqObject`** attribute.

Table 2-22 SearchRequest details

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	Not available in this release.
`reqObject`	**`qipws:VQIP_BASE_ENTITY`**	R	None	Object being search for and its search criteria

Request structure - SequenceAssociatedComponentsRequest message

Overview

The following request structure sequences a list of components associated with the infrastructure type. This is a specialized call which only has meaning for a few infrastructure types. For example, use it to change the sequence of UDA and UDA groups associated with the specified infrastructure type.

Table 2-23 SequenceAssociatedComponentsRequest details

Part name	Type	Opt / Req	Default	Description
commonInfo	qipws: CommonInfo	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
key	qipws: VQIP_KEY_ENTITY	R	None	Key identifying infrastructure type.
components	Qipws: LIST_BASE_ENTITY	R	None	List of associations in their new sequence.

Request structure - UpdateRequest message

Overview

The following common **`UpdateRequest`** structure is used for Update transactions. The northbound system uses the appropriate data type to populate the **`reqObject`** field. For example, if the northbound system wants to update a rule, an object of type **`RULE_REC`** is used to populate the **`reqObject`** field.

Table 2-24 UpdateRequest structure

Part name	Type	Opt/Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	Not available in this release.
`reqObject`	**`qipws:VQIP_BASE_ENTITY`**	R	None	Object being updated. In case the key of the object is being updated, **`reqObject`** includes the updated value of the key attribute.
`key`	**`qipws:VQIP_KEY_ENTITY`**	O	None	Key used to retrieve the object being updated.

Response structures

Overview

There are several types of response structures that are supported during normal processing, when there is no error. These messages are **`CountAssociationsResponse`**, **`FindAssociationResponse`**, **`Response`**, **`SearchResponse`**, and **`PageSearchResponse`**. These messages are described in the following paragraphs.

CountAssociationsResponse message

Response sent by VitalQIP to northbound system when count associations operation is successful.

Table 2-25 CountAssociationsResponse structure

Part name	Type	Opt/ Req	Default	Description
`result`	**`qipws:RESULT_TYPE`**	R	None	Status of the request.
`count`	**`xsd: int`**	R	None	Integer count of number of associations.

FindAssociationsResponse message

Response sent by VitalQIP to northbound system when find associations operation is successful.

Table 2-26 FindAssociationsResponse structure

Part name	Type	Opt/ Req	Default	Description
`result`	**`qipws:RESULT_TYPE`**	R	None	Status of the request.
`objects`	**`qipws: VQIP_KEY_ENTITY[ ]`**	R	None	Sequence of one or more VitalQIP supported keys that identify objects.

GetAssociatedComponentsResponse

Response sent by VitalQIP to northbound system when **`VQIPManager_GetAssociatedComponentsRequest`** operation is successful.

Table 2-27 GetAssociatedComponentsResponse structure

Part name	Type	Opt/ Req	Default	Description
result	**qipws:RESULT_TYPE**	R	None	Status of the request.
items	**qipws:LIST_BASE_ENTITY**	R	None	List of associations.

Response message

Successful response sent by VitalQIP to northbound system in response to most requests.

Table 2-28 Successful reponse

Part name	Type	Opt/ Req	Default	Description
result	**qipws:RESULT_TYPE**	R	None	Status of the request.
responseData	**qipws:RESPONSE_DATA**	O	None	Includes details regarding the status of a scheduled request.

SearchResponse message

Response sent by VitalQIP to northbound system when search operation is successful.

Table 2-29 Successful search response

Part name	Type	Opt/ Req	Default	Description
result	**qipws:RESULT_TYPE**	R	None	Status of the request.
objectData	**qipws: VQIP_BASE_ENTITY[ ]**	R	None	Sequence of one or more VitalQIP supported entities containing data corresponding to search request criterion.

PageSearchResponse message

Response sent by VitalQIP to northbound system when PageSearch operation is successful.

Table 2-30 Successful page search response

Part name	Type	Opt/ Req	Default	Description
result	**qipws:RESULT_TYPE**	R	None	Status of the request.

Part name	Type	Opt/ Req	Default	Description
objectData	**qipws:VQIP_BASE_ENTITY[ ]**	R	None	Sequence of one or more VitalQIP supported entities containing data corresponding to search request criterion.
totalPages	**xsd:int**	R	None	Total pages containing records that match the selection criteria.
currentPage	**xsd:int**	R	None	Index into page being returned.

Errors

Overview

In case of errors, a fault is thrown using the **`ApplFaultException`** exception. The exception contains **`ApplFault`** structure with the details of the errors

ApplFault

This is used to specify an application error from VitalQIP. The following table lists the attributes included in **`ApplFault`** structure.

Table 2-31 ApplFault structure

Part Name	Type	Description
`errorMsg`	**`xsd:string`**	Error message for the error encountered in VitalQIP.
`errorKey`	**`xsd:string`**	Key for the error encountered in VitalQIP.

Scheduling operations

Overview

Certain operations can be scheduled to allow the northbound system the ability to specify jobs to be run at a later time. As of Release 7.2, the following operations can be scheduled:

- V6 Subnet - Add operation.
- V6 Subnet - Split operation.
- V6 Subnet – Renumber operation
- Block – Renumber operation.

To schedule a request, the **`JOB_REQUEST`** object included in the relevant request message should be populated as part of the incoming request. For example, to add a V6 Subnet, Request object is populated. The response received from VitalQIP includes information in **`RESPONSE_DATA`** in the **`Response`** message. Please refer to "JOB_REQUEST" (p. 3-16) and "RESPONSE_DATA" (p. 3-25) for further details.

Access Control Devices

Overview

Purpose

The user can add, delete, get, search and update **`AcDevices`** using the following commands:

Table 2-32 AcDevices commands

Action	Operation	Incoming message	OutGoing message
Add	VQIPManager_AddRequest	AddRequest	Response
Delete	VQIPManager_DeleteRequest	DeleteRequest	Response
Get	VQIPManager_GetRequest	GetRequest	GetResponse
Search	VQIPManager_SearchRequest	SearchRequest	SearchResponse
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

Add, Delete, Get, Search and Update messages

The northbound system invokes VQIPManager_AddRequest for Add operations, VQIPManager_DeleteRequest for Delete operations, VQIPManager_GetRequest for Get operations, VQIPManager_SearchRequest for Search operations and VQIPManager_UpdateRequest for Update operations using the AC_DEVICE_REC structure. Details regarding required and optional attributes are given in "AC_DEVICE_REC attributes" (p. 2-37).

A Get operation retrieves a single record, while the search operation should be used to retrieve a list of one or more records. For the Search operation, all the command attributes are optional. Leaving them all unspecified returns all devices belonging to the current organization. A wildcard '*' character can also be used in any of the string fields.

Note: For an Update operation, all record data fields should be provided in the input file. Fields for which values are being nulled should be sent in as the attribute name with no values associated with it.

AC_DEVICE_REC attributes

The following table lists the required and optional attributes for operations on Access Control device records.

Table 2-33 AC_DEVICE_REC attributes

macAddress

Type:	**`xsd:string`**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	R	R	O	R
Description:	AcDevice MAC address.				

acSubscriber

Type:	**`xsd:string`**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	O	O
Description:	AcSubscriber login name with whom AcDevice is associated.				

userClass

Type:	**`xsd:string`**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	O	O
Description:	Userclass associated with the AcDevice (if it has a device level User Class).				

optionalAttributeList

Type:	**`qipws:UdaList`**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	O	O
Description:	List of User Defined Attributes and their groups for AcDevice. Refer to "UDALIST" (p. 3-40) for specification of UdaList.				

hostname

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	N/A	N/A	N/A	O	N/A
Description:	Host Name of the object (if the AcDevice has acquired a lease).				

domain

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	N/A	N/A	N/A	O	N/A
Description:	Domain Name of the object (if the AcDevice has acquired a lease).				

address

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	N/A	N/A	N/A	O	N/A
Description:	V4 address of the object (if the AcDevice has acquired a lease).				

disableupdates

Type:	**xsd:boolean**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	O	N/A	N/A	O
Description:	Do not send any MAC->User class updates to DHCP. Useful for bulk updates or if the user plans to push at the end of a series of operations. If passed on the command line, the presence of this option is taken as a Boolean flag to trigger the disable updates action. Omitting it, implies the cache will be updated each time an Add, Update or Delete event occurs.				

AC_DEVICE_KEY attributes

This structure is used to identify AcDevice for Update and set UDA operations.

Part Name	Type	Opt/Req	Description
macAddress	**xsd:string**	R	MAC Address.

Access Control Subscribers

Overview

Purpose

The user can add, delete, get, search and update **`AcSubscribers`**, and retrieve a list of all userClasses associated with an **`AcSubscriber`** (including those of any **`AcDevices`** also associated with the **`AcSubscriber`**) by using the following commands:

Table 2-34 AcSubscribers commands

Action	Operation	Incoming message	Outgoing message
Add	VQIPManager_AddRequest	AddRequest	Response
Delete	VQIPManager_DeleteRequest	DeleteRequest	Response
Get	VQIPManager_GetRequest	GetRequest	GetResponse
Search	VQIPManager_SearchRequest	SearchRequest	SearchResponse
Update	VQIPManager_UpdateRequest	UpdateRequest	Response
List	VQIPManager_GetRequest	GetRequest	GetResponse

Add, Delete, Get, Search, Update and List messages

The northbound system invokes VQIPManager_AddRequest for Add operations, VQIPManager_DeleteRequest for Delete operations, VQIPManager_GetRequest for Get and List operations, VQIPManager_SearchRequest for Search operations and VQIPManager_UpdateRequest for Update operations using the AC_SUBSCRIBER_REC structure. Details regarding required and optional attributes are given in "AC_SUBSCRIBER_REC attributes" (p. 2-41).

A Get operation retrieves a single record, while the search operation should be used to retrieve a list of one or more records. For the Search operation, all the command attributes are optional. Leaving them all unspecified returns all devices belonging to the current organization. A wildcard '*' character can also be used in any of the string fields.

Note: For an Update operation, all record data fields should be provided in the input file. Fields for which values are being nulled should be sent in as the attribute name with no values associated with it.

AC_SUBSCRIBER_REC attributes

The following table lists the required and optional attributes for operations on subscriber record attributes.

Table 2-35 AC_SUBSCRIBER_REC attributes

AcSubscriber

Type:	**`xsd:string`**				
Operation:	Add	Delete	Get/List	Search	Update
Optional/Required:	R	R	R	O	R
Description:	AcSubscriber username.				

AcSubscriberFirstName

Type:	**`xsd:string`**				
Operation:	Add	Delete	Get/List	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	AcSubscriber first name.				

AcSubscriberLastName

Type:	**`xsd:string`**				
Operation:	Add	Delete	Get/List	Search	Update
Optional/Required:	O	N/A	N/A	O	O
Description:	AcSubscriber last name.				

userClass

Type:	**`xsd:string`**				
Operation:	Add	Delete	Get/List	Search	Update
Optional/Required:	O	N/A	N/A	O	O
Description:	AcSubscriber DHCP user class name (if it has a subscriber level User Class).				

acSubscriberPassword

Type:	**xsd:string**				
Operation:	Add	Delete	Get/List	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	AcSubscriber encrypted password.				

maxDevices

Type:	**xsd:int**				
Operation:	Add	Delete	Get/List	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Maximum number of devices that can be managed by this AcSubscriber.				

acSubscriberStatus

Type:	**qipws:AC_SUBSCRIBER_STATUS_TYPE**				
Operation:	Add	Delete	Get/List	Search	Update
Optional/Required:	R	N/A	N/A	O	R
Description:	AcSubscriber status. Possible values include: Active, Inactive, Pending.				

optionalAttributeList

Type:	**qipws:UdaList**				
Operation:	Add	Delete	Get/List	Search	Update
Optional/Required:	O	N/A	N/A	O	O
Description:	List of User Defined Attributes and their groups for AcSubscribers. Refer to "UDALIST" (p. 3-40) for specification of UdaList.				

disableupdates

Type:	**xsd:boolean**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	O	N/A	N/A	O
Description:	Do not send any MAC->User class updates to DHCP. Useful for bulk updates or if the user plans to push at the end of a series of operations. If passed on the command line, the presence of this option is taken as a Boolean flag to trigger the disable updates action. Omitting it, implies the cache will be updated each time an Add, Update or Delete event occurs.				

AC_SUBSCRIBER_KEY attributes

The following table lists the required and optional attributes for operations on subscriber key attributes.

Table 2-36 AC_SUBSCRIBER_KEY attributes

This structure is used to identify AcSubscriber for Update and set UDA operations.

Part Name	Type	Opt/Req	Description
acSubscriber	**xsd:string**	R	AcSubscriber username.

Address pools

Overview

Purpose

The Address Allocation module in VitalQIP groups address blocks into “buckets” known as pools. Pools conform to a tree structure, starting at the root pool, known as the seed pool The pools at all other levels are referred to as child pools, or simply as pool. The following operations are allowed on both seed and child pools.

- Add
- Delete
- Get
- Get Children
- Merge
- Search
- Update

For operations on seed pools the **`SEED_POOL_REC`** structure is used, whereas the **`CHILD_POOL_REC`** structure is used for operations on child pools.

Address pools operation messages

Required messages for each operation are described in the following table.

Table 2-37 Address pools operation messages

Action	Operation	Incoming message	Outgoing message
Add	VQIPManager_AddRequest	AddRequest	Response
Delete	VQIPManager_DeleteRequest	DeleteRequest	Response
Get	VQIPManager_GetRequest	GetRequest	GetResponse
Get Children Pools	VQIPManager_GetContainedObjects Request	GetContainedObjectsRequest	GetContained ObjectsResponse
Merge	VQIPManager_MergeRequest	MergeRequest	Response
Paginated Search	VQIPManager_PageSearchRequest	PageSearchRequest	PageSearch Response
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

SEED_POOL_REC attributes

The following table lists the required and optional attributes for operations on seed pools.

Table 2-38 SEED_POOL_REC attributes

allowPendingStatus

Type:	**xsd:boolean**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O	N/A	N/A	N/A	N/A	O
Description:	Allow Pending Status On Blocks. Applicable only if pool is associated with a registry.					

blockAddedNotifyfreePendingBlockWhen

Type:	**xsd:boolean**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O	N/A	N/A	N/A	N/A	O
Description:	Send notification to contact when block is added to this pool.					

Type:	**qipws:FREE_PENDING_BLOCK_TYPE**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O*	N/A	N/A	N/A	N/A	O*
Description:	Used to inidicate when a pending block should be changed from “pending” to “free” status. Applicable only if **allowPendingStatus** is set to true. Required if **allowPendingStatus** is set to true.					

childAllocationNotify

Type:	**xsd:boolean**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O	N/A	N/A	N/A	N/A	O
Description:	Send notification to contact when block is allocated to child pool from this pool.					

contact

Type:	**xsd:string**					
Operation:	Add	Delete	Get	Merge	Search	Update

Optional/Required:	R	N/A	N/A	N/A	N/A	R
Description:	Comma separated list of e-mail addresses of the contacts for this pool.					

freePendingBlockDays

Type:	**xsd:short**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O*	N/A	N/A	N/A	N/A	O*
Description:	Number of days after which to change a pending block from "pending" status to "free" status. Applicable only when **freePendingBlockWhen** is set to **FREE_AFTER_DAYS**. Required when **freePendingBlockWhen** is set to **FREE_AFTER_DAYS**. Valid range: 0–30.					

name

Type:	**xsd:string**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	R	R	R	R	R	R
Description:	Name of up to 64 alphanumeric characters. For search operation, value contains search expression. Value of "*" indicates that all records that meet the specified criteria must be returned.					

parentName

Type:	**xsd:string**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O	O	O	O	O	O
Description:	Name of parent pool. Set to null.					

registry

Type:	**qipws:REGISTRY_TYPE**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O	N/A	N/A	N/A	N/A	O
Description:	Source for seed pool address. It cannot be changed for Modify operation.					

subnetOrg

Type:	**xsd:string**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O	N/A	N/A	N/A	N/A	O
Description:	***Used for IPv4 Only.*** Name of subnet organization to be used when a subnet/network is created in VitalQIP. If none is specified, look at parent pool's subnet organization.					

v4Algorithm

Type:	**qipws:ALLOCATION_ALGORITHM_TYPE**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	R	N/A	N/A	N/A	N/A	R
Description:	Allocation algorithm applicable to all IPv4 address blocks in this pool.					

v4HDThreshold

Type:	**xsd:short**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	R	N/A	N/A	N/A	N/A	R
Description:	When the Host Density (HD) ratio reaches this ratio, the user will be notified. Either v4 or v6 field is required. Valid range: 1-99.					

v4MaxAllocation

Type:	**xsd:short**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	R	N/A	N/A	N/A	N/A	R
Description:	Maximum number of IPv4 block allocation requests allowed from the child pool. Allocation requests from child pool cannot exceed this limit. Either v4 or v6 field is required. Valid range: 8-32.					

v4MinimumSparseSize

Type:	**xsd:short**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O*	N/A	N/A	N/A	N/A	O*
Description:	Minimum Block Size – Required if **v4Algorithm** is "SPARSE_ALLOCATION". Valid range: 8-32.					

v4ReverseZoneTemplate

Type:	**`xsd:string`**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O	N/A	N/A	N/A	N/A	O
Description:	Name of reverse zone template for creating reverse zone when a network/subnet is created (specifies primary and secondary DNS servers).					

v6Algorithm

Type:	**`qipws:ALLOCATION_ALGORITHM_TYPE`**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	R	N/A	N/A	N/A	N/A	R
Description:	Allocation algorithm applicable to all IPv6 address blocks in this pool.					

v6MaxAllocation

Type:	**`xsd:short`**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	R	N/A	N/A	N/A	N/A	R
Description:	Maximum number of IPv6 block allocation requests allowed from the child pool. Allocation requests from child pool cannot exceed this limit. Either v4 or v6 field is required. Valid range: 8-64.					

v6MinimumSparseSize

Type:	**`xsd:short`**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O*	N/A	N/A	N/A	N/A	O*
Description:	Minimum Block Size – Required if **`v6Algorithm`** is “SPARSE_ALLOCATION”. Valid range: 8-32.					

optionalAttributeList

Type:	**`qipws:UdaList`**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O	N/A	N/A	N/A	N/A	O
Description:	List of User Defined Attributes and their groups for pool. Refer to “UDALIST” (p. 3-40) for specification of UdaList.					

CHILD_POOL_REC attributes

The following table lists the required and optional attributes for operations on child pools.

Table 2-39 CHILD_POOL_REC attributes

name

Type:	**`xsd:string`**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	R	R	R	R		R
Description:	Name of up to 64 alphanumeric characters. For search operation, value contains search expression. Value of "*" will indicate that all records that meet the specified criteria must be returned.					

parentName

Type:	**`xsd:string`**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	R	R	R	R		R
Description:	Name of parent pool.					

grandParentName

Type:	**`xsd:string`**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	R	N/A	N/A	N/A		N/A
Description:	Name of the grandparent pool. For Add operations, **`parentName`** and **`grandParentName`** are used to uniquely identify the parent pool.					

contact

Type:	**`xsd:string`**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	R	N/A	N/A	N/A		R
Description:	Comma separated list of e-mail addresses of the contacts for this pool.					

recurseRequest

Type:	**`xsd:boolean`**

Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O	N/A	N/A	N/A		O
Description:	If true, this pool will request additional addresses from the parent if it cannot meet a request from its child.					

recurseRequestNotify

Type:	**xsd:boolean**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O	N/A	N/A	N/A		O
Description:	If true, sends notification to the contact when address request is made to the parent pool from this pool.					

childAllocationNotify

Type:	**xsd:boolean**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O	N/A	N/A	N/A		O
Description:	Send notification to contact when block is allocated to child pool from this pool.					

subnetOrg

Type:	**xsd:string**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O	N/A	N/A	N/A		O
Description:	***Used for IPv4 only.*** Name of subnet organization to be used when a subnet/network is created in VitalQIP. If none is specified, look at parent pool's subnet organization.					

v4Algorithm

Type:	**qipws:ALLOCATION_ALGORITHM_TYPE**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	R	N/A	N/A	N/A		R
Description:	Allocation algorithm applicable to all IPv4 address blocks in this pool.					

v4MinimumSparseSize

Type:	**xsd:short**

Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O*	N/A	N/A	N/A		O*
Description:	Minimum Block Size – Required if **`v4Algorithm`** is "SPARSE_ALLOCATION". Valid range: 8-32.					

v4Rule

Type:	**`xsd:string`**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O	N/A	N/A	N/A		O
Description:	Name of IPv4 allocation rule applicable to this pool.					

v4InheritReverseZoneTemplate

Type:	**`xsd:boolean`**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	R	N/A	N/A	N/A		R
Description:	If set to true, reverse zone template for this pool is inherited from parent pool.					

v4ReverseZoneTemplate

Type:	**`xsd:string`**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O	N/A	N/A	N/A		O
Description:	Name of reverse zone template for creating reverse zone when a network/subnet is created (specifies primary and secondary DNS servers). Not applicable when **`v4InheritReverseZoneTemplate`** is set to true. Required when **`v4InheritReverseZoneTemplate`** is not set to true.					

v4MaxAllocation

Type:	**`xsd:short`**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	R	N/A	N/A	N/A		R
Description:	Maximum number of IPv4 block allocation requests allowed from the child pool. Allocation requests from child pool cannot exceed this limit. Either v4 or v6 field is required. Valid range: 8-32.					

v4HDThreshold

Type:	**xsd:short**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	R	N/A	N/A	N/A		R
Description:	When the Host Density (HD) ratio reaches this ratio, the user will be notified. Either v4 or v6 field is required.Valid range: 1-99.					

v6Algorithm

Type:	**qipws:ALLOCATION_ALGORITHM_TYPE**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	R	N/A	N/A	N/A		R
Description:	Allocation algorithm applicable to all IPv6 address blocks in this pool.					

v6MinimumSparseSize

Type:	**xsd:short**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O*	N/A	N/A	N/A		O*
Description:	Minimum Block Size – Required if **v6Algorithm** is "SPARSE_ALLOCATION". Valid range: 8-32.					

v6Rule

Type:	**xsd:string**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O	N/A	N/A	N/A		O
Description:	Name of IPv6 allocation rule applicable to this pool.					

v6MaxAllocation

Type:	**xsd:short**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	R	N/A	N/A	N/A		R
Description:	Maximum number of IPv6 block allocation requests allowed from the child pool. Allocation requests from child pool cannot exceed this limit. Either v4 or v6 field is required. Valid range: 8-64.					

v4HDThreshold

Type:	**`xsd:short`**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	R	N/A	N/A	N/A		R
Description:	When the Host Density (HD) ratio reaches this ratio, the user will be notified. Either v4 or v6 field is required. Valid range: 1-99.					

v4HDThreshold

Type:	**`xsd:short`**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	R	N/A	N/A	N/A		R
Description:	When the Host Density (HD) ratio reaches this ratio, the user will be notified. Either v4 or v6 field is required. Valid range: 1-99.					

optionalAttributeList

Type:	**`qipws:UdaList`**					
Operation:	Add	Delete	Get	Merge	Search	Update
Optional/Required:	O	N/A	N/A	N/A		O
Description:	List of User Defined Attributes and their groups for pool. Refer to "UDALIST" (p. 3-40) for specification of **`UdaList`**.					

Add pool

Overview

The northbound system invokes **`VQIPManager_AddRequest`** operation for adding a pool in VitalQIP. This operation uses AddRequest message with **`SEED_POOL_REC`** structure to add a seed pool and **`CHILD_POOL_REC`** structure to add a child pool. Details regarding required and optional attributes are given in Table 2-38, “SEED_POOL_REC attributes” (p. 2-45) and Table 2-39, “CHILD_POOL_REC attributes” (p. 2-49).

Delete operation

Overview

The northbound system invokes **`VQIPManager_DeleteRequest`** operation to delete a pool from VitalQIP. This operation uses DeleteRequest with **`SEED_POOL_REC`** structure to delete a seed pool and **`CHILD_POOL_REC`** structure to delete a child pool. Details regarding required and optional attributes are given in Table 2-38, “SEED_POOL_REC attributes” (p. 2-45) and Table 2-39, “CHILD_POOL_REC attributes” (p. 2-49).

Deleting a seed pool removes the pool and all children pools, if any. A Delete operation can only be performed if all blocks under the pool (anywhere under the hierarchy) are “free”.

Get Children Pools

Overview

The northbound system invokes **`VQIPManager_GetContainedObjectsRequest`** operation for retrieving a list of child pools of a particular seed pool or a child pool. This operation uses **`GetContainedObjectsRequest`** Message with **POOL_KEYS** structure and it returns **`GetContainedObjectsResponse`** with a list of child pools.

GetContainedObjectsRequest message

The following table describes **`GetContainedObjectsRequest`** message.

Table 2-40 GetContainedObjectsRequest messages

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Please refer to "CommonInfo" (p. 2-7) for details about the **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	N/A in this release
`parent`	**`qipws:VQIP_KEY_ENTITY`**	R	None	POOL_KEYS structure identifying the pool to get the children of
`containedObjectType`	**`qipws:ENTITY_TYPE`**	R	None	Entity type to retrieve. The current release supports only following entity type: CHILD_POOL - get list of child pools.

GetContainedObjectsResponse message

Response sent by VitalQIP to northbound system when get children pool operation is successful.

Table 2-41 GetContainedObjectsResponse message

Part Name	Type	Opt/ Req	Default	Description
`result`	**`qipws:RESULT_TYPE`**	R	None	Status of the request.

Part Name	Type	Opt/ Req	Default	Description
containedObjects	**qipws: VQIP_KEY_ENTITY[ ]**	R	None	Sequence of one or more **POOL_KEYS** identifying the child pools of the sepecified pool.

Get pool

Overview

The northbound system invokes **`VQIPManager_GetRequest`** operation for retrieving a specific pool. This operation uses **`GetRequest`** structure with **`SEED_POOL_REC`** structure for seed pool and **`CHILD_POOL_REC`** for child pool. If the specified pool is found, attributes of the pool are returned. If the retrieved pool is a seed pool, **`SEED_POOL_REC`** structure containing pool attributes are returned in the **`Response`** message. If the retrieved pool is a child pool, **`CHILD_POOL_REC`** structure containing pool attributes are returned in the **`Response`** message. If the pool is not found, an empty record is returned in the **`Response`** message.

Merge pools

Overview

The northbound system invokes **`VQIPManager_MergeRequest`** operation for merging two pools. This operation uses **`MergeRequest`** message. The details of this message are given in "Request structure - MergeRequest message" (p. 2-25). The merge action moves all the blocks and child pools contained in the source pool into the target pool, and deletes the source pool. Both source and target pools must belong to the same Internet Registry, and both pools must have the same parent.

Search pool

Overview

The northbound system invokes **VQIPManager_PageSearchRequest** operation using **SEARCH_POOL_REC** structure for retrieving a list of one or more pool records matching the partial or full name of the pool. This operation uses **PageSearchRequest** message with **SEARCH_POOL_REC** structure. Details regarding required and optional attributes of **SEARCH_POOL_REC** in **PageSearchRequest** are given in the following table.

Table 2-42 SEARCH_POOL_REC attributes

Part name	Type	Opt/ Req	Default	Description
name	**xsd:string**	R	None	Partial or full name of the pools to be searched. "*" can be used at the end of name string to search for all pools starting with the specified prefix. "*" by itself can used to retrieve all pools in an organization.

Update operation

Overview

The northbound system invokes **`VQIPManager_UpdateRequest`** operation to update attributes of a pool. This operation uses an **`UpdateRequest`** message with **`SEED_POOL_REC`** structure to update a seed pool and **`CHILD_POOL_REC`** structure to a update child pool. Refer to the "Request structure - UpdateRequest message" (p. 2-30) for more information.

Please note that for Update operation, all data fields for a record, those that are being updated as well as those that are not being updated, should be sent in as input. Fields for which values are being nulled will be sent in as the attribute name with no values associated with it. Additionally, an Update operation cannot be used for modifying the registry with which the seed pool is associated.

Address block

Overview

Purpose

An address block is a continuous range of IP addresses that could be a subnet that uses CIDR notation for IPv4, or a typical IPv6 subnet. A seed block is an address block at the highest level managed by VitalQIP, and typically corresponds to the address space allocation from an RIR/LIR. The following operations are supported for blocks.

Address block (all non-seed blocks)

- Allocate to pool
- Expand
- Free
- Get
- Merge
- Move to another pool
- Renumber
- Return to parent
- Search
- Split
- Update

Seed block

- Add to pool
- Delete from pool
- Get
- Merge
- Move to another pool
- Search
- Update

For operations on seed blocks, **`SEED_BLOCK_REC`** structure is used; for operations on all other blocks, **`ADDRESS_BLOCK_REC`** structure is used.

Address block operation messages

Required messages each operations are described in the following table.

Table 2-43 Address block operations and messages

Action	Operation	Incoming message	Outgoing message
Add to Pool	VQIPManager_AddSeedBlockToSeedPool Request	AddSeedBlockToSeed PoolRequest	Response
Normal Block Allocation	VQIPManager_AllocateBlockToPool Request	AllocateBlockToPool MessageRequest	BlockAllocResponse
Delete from Pool	VQIPManager_DeleteSeedBlockFromSeed PoolRequest	DeleteSeedBlockFrom SeedPoolRequest	Response
Expand	VQIPManager_ExpandRequest	ExpandRequest	Response
Explicit Block Alloation	VQIPManager_ExplicitBlockAllocRequest	ExplicitBlockAllocMe ssageRequest	BlockAllocResponse
Free	VQIPManager_FreeRequest	FreeRequest	Response
Get	VQIPManager_GetRequest	GetRequest	GetResponse
Merge	VQIPManager_MergeRequest	MergeRequest	Response
Move	VQIPManager_MoveRequest	MoveRequest	Response
Paginated Search	VQIPManager_PageSearchRequest	PageSearchRequest	PageSearchResponse
Quick Block Allocation	VQIPManager_QuickBlockAllocRequest	QuickBlockAllocMess ageRequest	BlockAllocResponse
Recover	VQIPManager_RecoverBlockRequest	RecoverBlockRequest	RecoverBlockRespo nse
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

Address block operations

Address block attributes

The following table lists the required and optional attributes of **ADDRESS_BLOCK_REC** for operations on address blocks.

Table 2-44 ADDRESS_BLOCK_REC attributes

startAddress

Type:	**xsd:string**		
Operation:	Modify	Expand, Get, Merge, Move, Renumber, ReturnToParent, Split, Update	Free
Optional/Required:	R	R	R
Description:	IP Address string.		

prefixLength

Type:	**xsd:short**		
Operation:	Modify	Expand, Get, Merge, Move, Renumber, ReturnToParent, Split, Update	Free
Optional/Required:	R	R	R
Description:	Prefix length. Valid range is: • 8-32 for IPv4 • 8-64 for IPv6		

addressType

Type:	**qipws:V4V6_TYPE**		
Operation:	Modify	Expand, Get, Merge, Move, Renumber, ReturnToParent, Split, Update	Free
Optional/Required:	O	O	O
Description:	Either an IPv4 or IPv6 address type. If not specified, it is derived from address.		

blockStatus

Type:	**qipws:BLOCK_STATUS_TYPE**		
Operation:	Modify	Expand, Get, Merge, Move, Renumber, ReturnToParent, Split, Update	Free

Optional/Required:	R	O	O
Description:	Status of the block. For Free operation, either **`blockStatus`** or **`pool`** must be specified.		

pool

Type:	**`qipws:POOL_KEYS`**		
Operation:	Modify	Expand, Get, Merge, Move, Renumber, ReturnToParent, Split, Update	Free
Optional/Required:	R	R	O
Description:	Parent pool of the block. For Free operation, either **`blockStatus`** or **`pool`** must be specified.		

parentAddressBlock

Type:	**`qipws:ADDRESS_BLOCK_KEYS`**		
Operation:	Modify	Expand, Get, Merge, Move, Renumber, ReturnToParent, Split, Update	Free
Optional/Required:	N/A	N/A	N/A
Description:	Parent block of this address block.		

optionalAttributeList

Type:	**`qipws:UdaList`**		
Operation:	Modify	Expand, Get, Merge, Move, Renumber, ReturnToParent, Split, Update	Free
Optional/Required:	O	N/A	N/A
Description:	List of User Defined Attributes and their groups for address block. Refer to "UDALIST" (p. 3-40) for specification of **`UdaList`**.		

Allocate block to child pool

Overview

The northbound system invokes **`VQIPManager_AllocateBlockToPoolRequest`** operation to automatically allocate an address block to an existing child pool. The allocation is based on the allocation rules associated with the requesting pool and the allocation algorithms applicable to the parent pool.

This operation uses **`AllocateBlockToPoolMessageRequest`** message. The details of this message are given in the following table.

Table 2-45 AllocateBlockToPoolMessageRequest message attributes

Part name	Type	Opt/ Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale.
jobReq	qipws:JOB_REQUEST	O	None	Not available in this release.
`parameters`	**`qipws:AllocateBlockToPool Request`**	R	None	Structure containing information for automatic block allocation. Refer to Table 2-46, "AllocateBlockToPoolReques t structure " (p. 66) for the details of the structure.

This operation involves multiple processing steps. After allocating a block to the requested pool, this operation creates appropriate infrastructure objects in VitalQIP for the allocated IPv4 block and then sends an RIR report if required. If RIR report information is not provided, an RIR report is not sent after the block allocation process. This operation returns **`BlockAllocResponse`** described in "BlockAllocResponse message" (p. 2-97).

Table 2-46 AllocateBlockToPoolRequest structure

Part name	Type	Opt/ Req	Default	Description
`pool`	**`qipws:POOL_ KEYS`**	R	None	Pool to which new block is to be allocated.
`addressType`	**`qipws:V4V6_ TYPE`**	R	None	Type of address block to allocate, IPv4 or IPv6.

Part name	Type	Opt/ Req	Default	Description
networkContact	**xsd:string**	R	None	***IPv4 only.*** E-mail address of the contact for the network for IPv4, if one is to be created in VitalQIP.
requestedBlock	**qipws:ADDRESS_ BLOCK_KEYS**	O	None	Source address block in parent pool to use for allocation.
ruleName	**xsd:string**	O	None	Name of the rule to use during allocation. The rule must exist for the desired organization.
subnetName	**xsd:string**	O	None	Name of corresponding subnet to be created in VitalQIP – if the Address Managed (AM) module is not enabled. If not specified, it defaults to pool name.
optionalAttributeList	**UdaList**	O	None	List of User Defined Attributes and their groups for address block. Refer to "UDALIST" (p. 3-40) for specification of **UdaList**.
rirReport	**qipws:RIR_ REPORT**	O	None	If an RIR report needs to be sent, **RIR_REPORT** structure should be populated with appropriate data for the desired SWIP or INETNUM report. If this parameter is not provided, an RIR report is not sent after block is allocated in VitalQIP system. See "RIR_REPORT" (p. 2-98) for details of RIR_REPORT.

Free pending address blocks

Overview

The northbound system invokes **`VQIPManager_FreePendingBlockRequest`** operation to check and free all pending blocks matching the user specified criteria. This operation uses **`FreePendingBlockRequest`** message, as detailed in the following table.

Table 2-47 FreePendingBlockRequest details

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale.
`jobReq`	**`qipws:JOB_REQUEST`**	N/A	None	Job Scheduling information. Not supported in this release.
`level`	**`FREE_PENDING_CHECK_TYPE`**	R	None	Specifies the level at which the pending blocks will be checked and freed.
`override`	**`xsd:boolean`**	O	false	Flag to indicate whether to override all of the configured criteria or not.
`block`	**`ADDRESS_BLOCK_KEYS`**	O	None	Identifies the block to check and free. Required and applicable only when level = BLOCK
`pool`	**`POOL_KEYS`**	O	None	Identifies the pool to check and free pending blocks for. Required and applicable only when level = POOL.

Explicit address block allocation to child pool

Overview

The northbound system invokes **`VQIPManager_ExplicitBlockAllocRequest`** operation to explicitly allocate an address block to child pool. This operation uses **`ExplicitBlockAllocMessageRequest`** message. The details of this message are given in the following table.

Table 2-48 ExplicitBlockAllocMessageRequest message attributes

Part name	Type	Opt/ Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale.
`jobReq`	**`qipws:JOB_REQUEST`**	N/A	None	Job Scheduling information. Not supported in this release.
`parameters`	**`qipws:ExplicitBlockAlloc Request`**	R	None	Structure containing information for automatic block allocation. Refer to Table 2-49, "ExplicitBlockAllocRequest structure" (p. 2-69) for the details of the structure.

After allocating a block to the requested pool, this operation creates appropriate infrastructure objects in VitalQIP for the allocated IPv4 block and then sends an RIR report if RIR is required. If an RIR report information is not provided, an RIR report is not sent after the block allocation process. This operation returns **`BlockAllocResponse`** described in "BlockAllocResponse message" (p. 2-97).

Table 2-49 ExplicitBlockAllocRequest structure

Part name	Type	Opt/ Req	Default	Description
`startAddress`	**`xsd:string`**	R	None	Starting address of the block to be allocated.
`prefixLength`	**`xsd:short`**	R	None	Length of the block to be allocated
`pool`	**`qipws:POOL_KEYS`**	R	None	Pool to which to allocate address block.

Part name	Type	Opt/ Req	Default	Description
blockStatus	**qipws:BLOCK_STATUS_TYPE**	R	None	Status of the block to be allocated.Valid Status: FREE, RESERVED, SITE, USED.
addressType	**qipws:V4V6_TYPE**	R	None	Type of address block, IPv4 or IPv6.
subnetCreationOption	**qipws:SUBNET_CREATION_OPTION_TYPE**	O*	None	This option determines how the block is associated with the existing/new subnet in VitalQIP. Required when **blockStatus** is "USED", N/A for all other **blockStatus** values.
addressTemplate	**xsd:string**	O*	None	Name of address template to use. Required when **blockStatus** = USED.
subnetName	**xsd:string**	O	None	Name of subnet. Applicable only when **blockStatus** = USED.
subnetProfileName	**xsd:string**	O	None	Name of subnet profile template. Applicable only when **blockStatus** is "USED". N/A for all other **blockStatus** values.

Part name	Type	Opt/ Req	Default	Description
networkContact	**xsd:string**	O*	None	E-mail address of the contact for the Network, if Network is to be created in VitalQIP. Required when **blockStatus** = SITE or USED. N/A for all other **blockStatus** values.
allocationRuleName	**xsd:string**	O*	None	Name of allocation Rule to be used. Required when **blockStatus** = SITE. N/A for all other **blockStatus** values.
useDefaultRuleSize	**xsd:boolean**	O	False	Use default size from rule associate with pool.
optionalAttributeList	**qipws:UdaList**	O	None	List of User Defined Attributes and their groups for address block. Refer to "UDALIST" (p. 3-40) for specification of **UdaList**.
rirReport	**qipws:RIR_REPORT**	O	None	If an RIR report needs to be sent, the **RIR_REPORT** structure should be populated with appropriate data for the desired SWIP or INETNUM report. If an RIR report is not required to be sent, or if this parameter is not provided, the RIR report is not sent after a block is allocated in the VitalQIP system. Refer to "RIR_REPORT" (p. 2-98) for details of **RIR_REPORT**.

Expand address block

Overview

The northbound system invokes **`VQIPManager_ExpandBlockRequest`** operation for expanding an address block. This operation uses **`ExpandBlockRequest`** message with **`ADDRESS_BLOCK_REC`** structure. The details of this message are given in "BlockAllocResponse message" (p. 2-97).

Table 2-50 ExpandBlockRequest details

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	Not available in this release.
`addressBlock`	**`qipws:ADDRESS_BLOCK_REC`**	R	None	An address block to expand.

Block expansion allows a block to acquire contiguous space from the parent pool. Expansion is allowed on all levels of blocks. Expansion is allowed for the block with status "USED", "FREE", "RESERVED" or "ORIGIN".

Free address block

Overview

The northbound system invokes **`VQIPManager_FreeBlockRequest`** for freeing an address block. This operation uses **`FreeBlockRequest`** message with **`ADDRESS_BLOCK_REC`** structure. The details of this message are given in the following table.

Table 2-51 FreeBlockRequest message attributes

Part name	Type	Opt/ Req	Default	Description
`commonInfo`	**`CommonInfo`**	R	None	Contains user organization and locale.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	Not available in this release.
`block`	**`ADDRESS_BLOCK_REC`**	R	None	Address block to be freed.
`returnTo Parent`	**`xsd:boolean`**	O	False	Flag to indicate whether block should be returned to parent. Default = false. If set to true, the block is returned to the parent after its status is changed to FREE.

Note: Freeing an address block is allowed only on a block with “USED” status.

Overview

The northbound system invokes **`VQIPManager_GetRequest`** operation to get a list of attributes for an address block. This operation uses **`GetRequest`** message with **`ADDRESS_BLOCK_REC`** structure. Refer to the table in “Address block attributes” (p. 2-64) for details of required elements of **`ADDRESS_BLOCK_REC`** structure for this operation.

Merge address block

Overview

The northbound system invokes **`VQIPManager_MergeRequest`** operation for merging two Address blocks. This operation uses **`MergeRequest`** message with **`ADDRESS_BLOCK_REC`** structure. The details of this message are given in section "Request structure - MergeRequest message" (p. 2-25).

The merge operation is allowed for the following block status values: "USED", "RESERVED", FREE". Note that the used block cannot have prior RIR reporting.

Both the blocks being merged must be siblings within the same pool. (If they are in different pools, the user can use the move block operation to get the blocks in the same pool.) Both blocks cannot have "FREE" status. Two blocks being merged with differing status values will have the following resulting status:

- "USED" + any status = "USED"
- "RESERVED" + "FREE" = "RESERVED"

If block is "USED", and if the subnet was created in VitalQIP, the size of the corresponding subnet in VitalQIP is also expanded.

Note: No RIR reporting occurs following the completion of the merge block.

Move to another pool

Overview

The northbound system invokes **VQIPManager_MoveRequest** operation for moving address block to a different pool. This operation moves the specified block, all of its children in the block hierarchy, and all of the child pools that contain segments of this block, to the destination pool. This operation uses **MoveRequest** message with **ADDRESS_BLOCK_REC** structure. The details of this message are given in the following table.

Table 2-52 MoveRequest message attributes

Part name	Type	Opt/ Req	Default	Description
CommonInfo	**qipws:CommonInfo**	R	None	Contains user organization and locale.
jobReq	**qipws:JOB_REQUEST**	O	None	Not available in this release.
source	**qipws:VQIP_BASE_ ENTITY**	R	None	**ADDRESS_BLOCK_REC** identifying the address block if moving a non-seed block, or **SEED_BLOCK_REC** identifying the seed address block if moving a seed block.
target	**qipws:VQIP_BASE_ ENTITY**	R	None	**POOL_REC** identifying the target pool.

For the “Move” action to be applicable, the following conditions must be met:

- The source and destination pools must be siblings.
- The block must be the highest level block in the pool.
- All address space in any child pool that contains a portion of the block to be moved must only contain address space of the block to be moved.

Quick block allocation to child pool

Overview

The northbound system invokes **`VQIPManager_QuickBlockAllocRequest`** operation to perform quick allocation to the specified child pool. If the specified pool does not already exist, the pool is first created before allocating the block. The block allocation is based on the specified rules and the allocation algorithms applicable to the parent pool.

This operation uses **`QuickBlockAllocMessageRequest`** message. The details of this message are given in the following table.

Table 2-53 QuickBlockAllocMessageRequest message attributes

Part name	Type	Opt/ Req	Default	Description
`commonInfo`	**`CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`JOB_REQUEST`**	O	None	Not available in this release.
`parameters`	**`qipws:QuickBlockAllocRequest`**	R	None	Structure containing information for block allocation. Refer to Table 2-54, "QuickBlockAllocRequest structure" (p. 2-76) for details of the structure.

This operation involves multiple processing steps. If the specified child pool does not already exist in VitalQIP, the pool is created before allocating a block to it. After allocating a block to the requested pool, this operation creates appropriate infrastructure objects in VitalQIP for the allocated IPv4 block and then sends an RIR report if an RIR required. If RIR report information is not provided, an RIR report is not sent after the block allocation process. This operation returns **`BlockAllocResponse`** described in "BlockAllocResponse message" (p. 2-97).

Table 2-54 QuickBlockAllocRequest structure

Part name	Type	Opt/ Req	Default	Description
`childPoolName`	**`xsd:string`**	R	None	Pool to which the new block is to be allocated.

Part name	Type	Opt/ Req	Default	Description
parentPool	**qipws:POOL_ KEYS**	R	None	Pool's parent pool.
addressType	**qipws:V4V6_ TYPE**	R	None	Type of address block to allocate, IPv4 or IPv6.
ruleName	**xsd:string**	R	None	Name of allocation rule to be used.
poolContact	**xsd:string**	R	None	E-mail address of the contact for the child pool, if one is to be created in VitalQIP.
networkContact	**xsd:string**	R	None	E-mail address of the contact for the IPv4 network.
subnetName	**xsd:string**	O	None	Name of corresponding subnet to be created in VitalQIP – if the Address Managed (AM) module is not enabled. If not specified, it defaults to the pool name.
childPoolOptional AttributeList	**qipws:UdaList**	O	None	List of User Defined Attributes and their groups for child pool. Applicable when the specified child pool needs to be created in VitalQIP. A child pool is created with the specified User Defined Attributes. Refer to "UDALIST" (p. 3-40) for specification of **UdaList**.
addressBlockOptional AttributeList	**qipws:UdaList**	O	None	List of User Defined Attributes and their groups for address block. Refer to "UDALIST" (p. 3-40) for specification of **UdaList**.

Part name	Type	Opt/ Req	Default	Description
rirReport	**qipws:RIR_REPORT**	O	None	If an RIR report needs to be sent, the RIR_REPORT structure should be populated with appropriate data for the desired SWIP or INETNUM report. If an RIR report is not required to be sent, or if this parameter is not provided, the RIR report is not sent after the block is allocated in the VitalQIP system. Refer to "RIR_REPORT" (p. 2-98) for details of RIR_REPORT.

Recover address blocks

Overview

The northbound system invokes **`VQIPManager_RecoverBlockRequest`** operation to recover a block in a pending state. After successful operation block status is changed from "Pending" back to "Used". This operation uses **`RecoverBlockMessageRequest`** Message. The details of this message are given in the following table.

Table 2-55 QuickBlockAllocMessageRequest message attributes

Part name	Type	Opt/ Req	Default	Description
`commonInfo`	**`CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`JOB_REQUEST`**	O	None	Not available in this release.
`parameters`	**`qipws:RecoverBlock Request`**	R	None	Structure containing information for block to recover Refer to Table 2-56, "RecoverBlockRequest structure" (p. 2-79) for details of the structure.

This operation involves multiple processing steps. After the requested block has been recovered, VitalQIP sends an RIR report if required. If RIR report information is not provided, then the RIR report is not sent after the block recover process. This operation returns RecoverBlockResponse, described in Table 2-57.

Table 2-56 RecoverBlockRequest structure

Part name	Type	Opt/ Req	Default	Description
`block`	**`qipws:ADDRESS _BLOCK_KEYS`**	R	None	Identifies the block to check and recover.

Part name	Type	Opt/ Req	Default	Description
networkContact	**xsd:string**	O	Value of blockNetworkContact from VitalQIP configuration file *$QIPHOME/web/ conf/ qip.properties*	E-mail address of the contact for the IPv4 network where the block is located. **Note:** This is a required parameter if the default value is not defined in the VitalQIP Web Service configuration file.
rirReport	**qipws:RIR_ REPORT**	O	None	If an RIR report needs to be sent, the RIR_REPORT structure should be populated with appropriate data for the desired SWIP or INETNUM report. If an RIR report is not required to be sent, or if this parameter is not provided, the RIR report is not sent after the block is allocated in the VitalQIP system. Refer to "RIR_REPORT" (p. 2-98) for details of RIR_REPORT.

A response is sent by VitalQIP to the northbound system when a block recover request is either partially successful or completely successful.

Table 2-57 RecoverBlockResponse structure

Part name	Type	Opt/ Req	Default	Description
result	**qipws:RESULT_ TYPE**	R	None	SUCCESS=Block is recovered and RIR report sent, if required and specified in input request. PARTIAL_FAILURE=Block has been recovered but RIR reporting failed.

Part name	Type	Opt/ Req	Default	Description
errorRecs	**qipws: PARTIAL_ FAILURE_REC**	O	None	RIR Reporting failure details. This is populated when **result=PARTIAL_ FAILURE**. Refer to **PARTIAL_FAILURE_REC** structure in "PARTIAL_FAILURE_REC" (p. 3-21).
blockKey	**qipws:ADDRESS _BLOCK_KEYS**	R	None	Identifies the recovered block.

Renumber block

Overview

The northbound system invokes **`VQIPManager_RenumberRequest`** operation for renumbering a block. The renumber block operation can be used to change the prefix of the address block, without having to change the IP address component to the right of the prefix (that is, the InterfaceId). This operation uses **`RenumberRequest`** message. The details of this message are given in the following table.

Table 2-58 RenumberRequest message attributes

Part name	Type	Opt/ Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	R	None	Job Scheduling Information. Refer to "JOB_REQUEST" (p. 3-16) for information on **`JOB_REQUEST`** structure.
`source`	**`qipws:VQIP_BASE_ENTITY`**	R	None	**`ADDRESS_BLOCK_REC`** structure identifying the address block to be renumbered.
`target`	**`qipws:VQIP_BASE_ENTITY`**	R	None	**`ADDRESS_BLOCK_REC`** structure identifying the target address block.

Note: This operation is allowed only on IPv6 address blocks with "USED" status and with no prior RIR reporting performed on them.

Return block to parent pool

Overview

The northbound system invokes **`VQIPManager_ReturnToParentBlockRequest`** operation for returning a free address block to the parent pool. This operation uses **`AddressBlockRequest`** message with **`ADDRESS_BLOCK_REC`** structure. The details of this message are given in "BlockAllocResponse message" (p. 2-97).

Table 2-59 ReturnToParentBlockRequest message attributes

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	R	None	Not available in this release.
`addressBlock`	**`qipws:ADDRESS_BLOCK_REC`**	R	None	Block to return.

For this action to be performed, the block has to be the highest-level block in the pool. When the block is returned to its parent pool, it is removed from the current pool and the corresponding block in the parent pool (that previously had a status of "ALLOCATED") is marked as "FREE".

Search blocks

The Search operation allows a user to find an address block by the name of the pool, block length as well as the status of the block. Furthermore, if User-Defined Attributes of pool and/or address block exist, the user can also enter the attribute names (partial or full) to search for the address block. The wildcard character for search operation is '*'.

The northbound system invokes **VQIPManager_PageSearchRequest** operation to get a list of address blocks matching the specified search criteria. This operation uses **PageSearchRequest** message. Refer to "Request structure - MoveObjectRequest message" (p. 2-26) for more information on **PageSearchRequest**. The **SEARCH_ADDRESS_BLOCK_REC** structure is used for searching address blocks matching the specified search criteria. The details of **SEARCH_ADDRESS_BLOCK_REC** structure are given in the following table.

Table 2-60 SEARCH_ADDRESS_BLOCK_REC structure

Part name	Type	Opt/ Req	Description
addressType	**qipws:V4V6_TYPE**	R	IPv4 or IPv6 address type.
startAddress	**xsd:string**	R	Partial or full IP address string
prefixLength	**xsd:short**	O	Prefix length. Valid ranges are: • 8-32 for IPv4 • 8-64 for IPv6
pool	**qipws:POOL_KEYS**	O	Parent pool of the block.
blockStatusList	**qipws:BLOCK_STATUS_TYPE**	O	List of block statuses for which to search.
poolOptional AttributeList	qipws:UdaList	O	List of pool UDAs and their groups. Refer to "UDALIST" (p. 3-40) for specification of **UdaList**
blockOptional Attributes	**qipws:UdaList**	O	List of UDAs and their groups for address block. Refer to "UDALIST" (p. 3-40) for specification of **UdaList**.

Split block

Overview

The northbound system invokes **`VQIPManager_SplitRequest`** operation for splitting an address block. The split operation creates multiple smaller blocks out of the given block. This operation uses **`SplitRequest`** message. The details of this message are given in the following table.

Table 2-61 SplitRequest message attributes

Part name	Type	Opt/ Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	Not supported in this release.
`source`	**`qipws:ADDRESS_ BLOCK_REC`**	R	None	Address block record that needs to be split.
`requestedSize`	**`xsd:short`**	R	None	Desired length of the new address blocks. The user can advance only by a maximum of 8 bits (that is, /40 can only be split up into /48s).

Note: This operation is allowed only on IPv6 address blocks with "USED" status and no prior RIR reporting done on them.

Update address block

Overview

The northbound system invokes **`VQIPManager_UpdateRequest`** operation with **`UpdateRequest`** message containing **`ADDRESS_BLOCK_REC`** structure to update an address block. Only the status field can be updated via an update operation on an address block. The status update of a "FREE" block to "RESERVED" or "RESERVED" block to "FREE" are allowed. No other changes are allowed.

Note: For an Update operation, all data fields for a record, those that are being updated as well as those that are not being updated should be sent in as input. Fields for which values are being nulled will be sent in as the attribute name with no values associated with it. Additionally, an Update operation cannot be used for modifying the registry with which the seed pool is associated.

Seed block operations

Seed block attributes

The following table lists the required and optional attributes of **SEED_BLOCK_REC** for operations on seed blocks.

Table 2-62 SEED_BLOCK_REC attributes

startAddress

Type:	**xsd:string**	
Operation:	Add To Seed Pool, Update	Delete From Seed Pool, Get, Merge, Move
Optional/Required:	R	R
Description:	IP Address string.	

prefixLength

Type:	**xsd:short**	
Operation:	Add To Seed Pool, Update	Delete From Seed Pool, Get, Merge, Move
Optional/Required:	R	R
Description:	Prefix length. Valid ranges are: • 8-32 for IPv4 • 8-64 for IPv6	

addressType

Type:	**qipws:V4V6_TYPE**	
Operation:	Add To Seed Pool, Update	Delete From Seed Pool, Get, Merge, Move
Optional/Required:	O	O
Description:	IPv4 or IPv6 address type. If not specified, it is derived from the address.	

blockStatus

Type:	**qipws:BLOCK_STATUS_TYPE**	
Operation:	Add To Seed Pool, Update	Delete From Seed Pool, Get, Merge, Move
Optional/Required:	N/A	N/A

Description:	Status of the block.

pool

Type:	**qipws:POOL_KEYS**	
Operation:	Add To Seed Pool, Update	Delete From Seed Pool, Get, Merge, Move
Optional/Required:	R	R
Description:	Parent pool of the block.	

parentAddressBlock

Type:	**qipws:ADDRESS_BLOCK_KEYS**	
Operation:	Add To Seed Pool, Update	Delete From Seed Pool, Get, Merge, Move
Optional/Required:	N/A	N/A
Description:	Not applicable for seed block.	

maintainer

Type:	**xsd:string**	
Operation:	Add To Seed Pool, Update	Delete From Seed Pool, Get, Merge, Move
Optional/Required:	O*	N/A
Description:	Name of maintainer responsible for RIR reporting on this block. Required if pool is associated with a registry.	

threshold

Type:	**xsd:short**	
Operation:	Add To Seed Pool, Update	Delete From Seed Pool, Get, Merge, Move
Optional/Required:	O	N/A
Description:	Host Density Threshold percentage. When the address utilization reaches this threshold, notifications will be sent. Valid range: 0-100.	

reportTypes

Type:	**qipws:RIR_REPORT_TYPE**

Operation:	Add To Seed Pool, Update	Delete From Seed Pool, Get, Merge, Move
Optional/Required:	O	N/A
Description:	List of report types for RIR Reporting. If not specified, the SWIP report for ARIN and INETNUM report for RIPE/APNIC are not generated.	

optionalAttributes

Type:	**`qipws:UdaList`**	
Operation:	Add To Seed Pool, Update	Delete From Seed Pool, Get, Merge, Move
Optional/Required:	O	O
Description:	List of User Defined Attributes and their groups for seed block. Refer to "UDALIST" (p. 3-40) for specification of **`UdaList`**.	

Add seed block to seed pool

Overview

The northbound system invokes **`VQIPManager_AddSeedBlockToSeedPoolRequest`** operation to add a new seed block to an existing seed pool. This operation uses **`AddSeedBlockToSeedPoolRequest`** message. The details of required and optional parameters of **`SEED_BLOCK_REC`** are given in "Seed block attributes" (p. 2-87). The details of this message are described in the following table.

Table 2-63 AddSeedBlockToSeedPoolRequest message attributes

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	Not available in this release.
`addressBlock`	**`qipws:ADDRESS_BLOCK_REC`**	R	None	Address block to add.

Delete seed block from seed pool

Overview

The northbound system invokes **`VQIPManager_DeleteSeedBlockFromSeedPoolRequest`** operation to delete a seed block from seed pool. This operation uses **`DeleteSeedBlockFromSeedPoolRequest`** message with **`SEED_BLOCK_REC`** structure. The details of required and optional parameters of **`SEED_BLOCK_REC`** are given in "Seed block attributes" (p. 2-87).

The details of this message are described in the following table.

Table 2-64 DeleteSeedBlockFromSeedPoolRequest message attributes

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`qipws:CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	Not available in this release.
`addressBlock`	**`qipws:ADDRESS_BLOCK_REC`**	R	None	Address block to delete.

Get seed block

The northbound system invokes **`VQIPManager_GetRequest`** operation to get a list of attributes of a seed block. This operation uses **`GetRequest`** message with **`SEED_BLOCK_REC`** structure. The details of required and optional parameters of **`SEED_BLOCK_REC`** are given in "Seed block attributes" (p. 2-87).

Merge seed blocks

Overview

The northbound system invokes **`VQIPManager_MergeRequest`** operation for merging two seed blocks. This operation uses **`MergeRequest`** message with **`SEED_BLOCK_REC`** structure. The details of **`MergeRequest`** are given in "Request structure - MergeRequest message" (p. 2-25). The details of required and optional parameters of **`SEED_BLOCK_REC`** are given in "Seed block attributes" (p. 2-87).

Move seed block to another pool

Overview

The northbound system invokes **`VQIPManager_MoveRequest`** operation for moving a seed block to a different seed pool. This operation uses **`MoveRequest`** message with **`SEED_BLOCK_REC`** structure. Refer to "Move to another pool" (p. 2-75) for details of **`MoveRequest`** message. The details of required and optional parameters of **`SEED_BLOCK_REC`** are given in "Seed block attributes" (p. 2-87).

Search seed block

Overview

Seed block is searched in the same way as an address block. Refer to "Search blocks" (p. 2-84) for details on searching blocks.

Update seed block

Overview

The northbound system invokes **`VQIPManager_UpdateRequest`** operation with **`SEED_BLOCK_REC`** to update a seed block. Only the status field and RIR report type list can be updated via an update operation on seed block. The status update of a "FREE" block to "RESERVED" or "RESERVED" block to "FREE" are allowed. No other changes are allowed.

Note: For Update operation, all data fields for a record, those that are being updated, as well as those that are not being updated, should be sent in as input. Fields for which values are being nulled are sent in as the attribute name with no values associated with it. Additionally, an Update operation cannot be used for modifying the registry with which the seed pool is associated.

BlockAllocResponse message

Overview

The **AddressBlockResponse** structure described in the following table is sent by VitalQIP to the northbound system when block allocation (automatic, explicit or quick) request is either partially successful or completely successful.

Table 2-65 AddressBlockResponse structure

Part name	Type	Opt/ Req	Default	Description
result	**qipws:RESULT_TYPE**	R	None	SUCCESS = Entire block allocation process was successful. PARTIAL_FAILURE = Block has been allocated to the child pool but either creation of infrastructure objects or RIR reporting or both failed. The failed step and the failure cause are returned in **errorRecs**.
errorRecs	**qipws:PARTIAL_FAILURE_REC**	O	None	Records with the details of the failure. This is populated when result = PARTIAL_FAILURE. Refer to **PARTIAL_FAILURE_REC** structure in "PARTIAL_FAILURE_REC" (p. 3-21).

RIR_REPORT

Overview

RIR_REPORT structure is used to specify data to be used for RIR Report during block allocation. The following table lists the required and optional attributes of RIR_REPORT structure.

Table 2-66 RIR_REPORT structure

Element name	Type	Opt/ Req	Default	Description
reportType	**qipws:RIR_REPORT_TYPE**	R	None	SWIP or INETNUM Report Type. Allowable values are: • ARIN-Reallocate • ARIN-Reassign-Detailed • ARIN-Reassign-Simple • ARIN-IPV6-Reallocate • ARIN-IPV6-Reassign • INETNUM for IPV4 • INETNUM for IPV6 • Do Not Generate Report
arinReport	**qipws:ARIN_RA_DETAILED_ REPORT**	O	None	Structure containing values for ARIN SWIP Templates. Required when **reportType** is one of the following: • ARIN-Reallocate • ARIN-Reassign-Detailed • ARIN-IPV6-Reallocate • ARIN-IPV6-Reassign Refer to "SWIP reporting" (p. 2-101) for details on SWIP reporting.

Element name	Type	Opt/ Req	Default	Description
arinReassign SimpleReport	**qipws:ARIN_RA_SIMPLE_ REPORT**	O	None	Structure containing values for ARIN SWIP Simple Template. Required when report type is ARIN-Reassign-Simple. Refer to "SWIP reporting" (p. 2-101) for details on SWIP reporting.
ripeReport	**qipws:RIPE_INETNUM_ REPORT**	O	None	Structure containing values for Ripe Inetnum Report. Required when seed pool configured with RIPE support is turned on and **reportType** is one of the following: • INETNUM for IPV4 • INETNUM for IPV6 Refer to "RIPE Reporting" (p. 2-116) for details on RIPE reporting.
apnicReport	**qipws:APNIC_INETNUM_ REPORT**	O	None	Structure containing values for APNIC Inetnum Report. Required when seed pool configured with APNIC support is turned on and **reportType** is one of the following: • INETNUM for IPV4 • INETNUM for IPV6 Refer to "APNIC Reporting" (p. 2-110) for details on APNIC reporting.

Element name	Type	Opt/ Req	Default	Description
afrinicReport	qipws:AFRINIC_INETNUM_ REPORT	O	None	Structure containing values for AfriNIC Inetnum Report. Required when seed pool configured with AfriNIC support is turned on and reportType is one of the following: • INETNUM for IPV4 • INETNUM for IPV6 Refer to "AfriNIC reporting" (p. 2-123) for details on AfriNIC reporting.

SWIP reporting

Overview

VitalQIP contains optional support for registering address space with ARIN. If a seed pool is configured with ARIN support turned on, any block allocation that creates a VitalQIP subnet anywhere below the seed pool in the hierarchy will generate a SWIP report. To generate a SWIP report, user must specify either **`arinReport`** or **`arinReassignSimpleReport`** elements. The specified information will be formatted into the selected SWIP template and e-mailed to the ARIN e-mail address specified on the seed pool.

VitalQIP does not generate SWIP reports for block allocations that are smaller than a /29 because ARIN does not wish to be notified about such block sizes.

For IPv4 address blocks, ARIN SWIP templates that are supported in VitalQIP include:

- ARIN-REALLOCATE-4.2
- ARIN-REASSIGN-SIMPLE-4.2
- ARIN-REASSIGN-DETAILED-4.2

For IPv6 address blocks, ARIN SWIP templates that are supported in VitalQIP include:

- ARIN-IPv6-REALLOCATE-4.2
- ARIN-IPv6-REASSIGN-4.2

For more information on these SWIP templates, please access www.arin.net.

Whenever a Free Block operation is performed on a block that contains a valid SWIP type and network name, a corresponding Free SWIP report is sent automatically.

- For a block allocated with an ARIN-REASSIGN-SIMPLE-4.2 template, it is freed by sending another ARIN-REASSIGN-SIMPLE-4.2 template and specifying R for the Registration Action in the template.
- For a block allocated with an ARIN-REASSIGN-DETAILED-4.2 template or an ARIN-REALLOCATE-4.2 template, it is freed by sending an ARIN-NET-MOD-4.2 template and specifying R for Registration Action in the template.

VitalQIP does not place any restrictions on the field lengths or the values entered into the fields. In some cases, some fields are not applicable if other fields have a value and some fields may have length or content limitations but VitalQIP will not enforce any restriction. This is by design since the format of the SWIP templates are subject to change by ARIN. Please refer to www.arin.net for more information on attributes in SWIP templates.

The following subsections contain details of the structures used to provide data for SWIP Templates.

ARIN_RA_SIMPLE_REPORT attributes

This structure is used to specify data to be used for SWIP report using ARIN-Reassign-Simple-4.2 template during IPv4 block allocation when seed pool is configured with ARIN support turned on.

VitalQIP Web Service can be configured to use the default values for customer information section in Arin Simple Report. If these default values are configured in VitalQIP Web Service configuration file located at *$QIPHOME/web/conf/qip.properties*, then the Web Service is populated the default customer information if it is not provided in the input.

The following table lists the required and optional attributes of **`ARIN_RA_SIMPLE_REPORT`** structure.

Table 2-67 ARIN_RA_SIMPLE_REPORT attributes

Element name	Type	Opt/ Req	Default	Description
`registration Action`	**`qipws:REGISTRATION_ACTION_TYPE`**	R	None	Registration Action.
`originAs`	**`xsd:string`**	O	None	List of comma separated AS numbers from which the network may originate.
`customerInfo`	**`qipws:INFO_REC`**	O	Default values defined in *$QIPHOME/web/conf/qip.properties*	Customer Information.
`usePrivateName`	**`xsd:Boolean`**	O	True or False	Indicates whether to use private customer name configured in VitalQIP as Customer Name in the report. If set to true, Customer Name field in RIR Report is populated with privateCustomerName property in *$QIPHOME/web/conf/qip.properties*.

Element name	Type	Opt/ Req	Default	Description
usePrivateAddress	**xsd:Boolean**	O	True or False	Indicates whether to use private address configured in VitalQIP as Customer Address in the report. If set to true, Customer Address field in RIR Report is populated with privateCustomerAddress property in *$QIPHOME/web/conf/qip.properties*.
publicComments	**xsd:string**	O	None	Public Comments.
customerCustomized Info	**xsd:string**	O	None	Spare field for user defined data to be included at the end of the report. Value of this field is included in the report as is.

ARIN_RA_DETAILED_REPORT attributes

This structure is used to specify data to be used for SWIP report using one of the following templates when a seed pool is configured with ARIN support turned on:

- ARIN-Reallocate-4.2 SWIP template for IPv4 address block
- ARIN-Reassign-Detailed-4.2 SWIP template for IPv4 address block
- ARIN-IPV6-Reallocate-4.2 SWIP template for IPv6 address block
- ARIN-IPV6-Reassign-4.2 SWIP template for IPv6 address block

The following table lists the required and optional attributes of **ARIN_RA_DETAILED_REPORT** structure.

Table 2-68 ARIN_RA_DETAILED_REPORT attributes

Element name	Type	Opt/ Req	Default	Description
downStreamOrgId	**xsd:string**	R*	None	ID of Downstream Organization to whom address is being reallocated or reassigned. *If Downstream ORG ID is provided, then **orgInfo** and **orgContactInfo** information is not needed.
orgInfo	**qipws:INFO_REC**	R	None	Organization information.
orgContactInfo	**qipws:CONTACT_INFO_REC**	R	None	Organization contact information. Information of the contact responsible for the administrative and technical operations of the organization.
networkContactInfo	**qipws:CONTACT_INFO_REC**	R	None	Network contact information. Information of contact responsible for the technical aspects of maintaining the IP address space.
originAs	**xsd:string**	O	None	List of comma separated AS numbers from which the network may originate.
dnsReverseMapping Nameserver1	**xsd:string**	O	None	Fully-qualified hostname of DNS Reverse Mapping Nameserver.
dnsReverseMapping Nameserver2	**xsd:string**	O	None	Fully-qualified hostname of DNS Reverse Mapping Nameserver.

Element name	Type	Opt/ Req	Default	Description
networkCustomizedInfo	**xsd:string**	O	None	Spare field for user defined data to be included at the end of Network section in the report. The value of this field is included at the end of Network section without any modification. This field should contain name value pair of user defined fields as they need to be included in the report.
netPOCType	**qipws:NET_POC_TYPE**	O	None	Define the function of the POC. Allowable values are: • T (for resource technical contact) • AB (for resource abuse contact) • N (for resource NOC contact) • T_AND_AB (for both resource technical contact and resource abuse contact) • T_AND_N (for both resource technical contact and resource NOC contact) • AB_AND_N (for both resource abuse contact and resource NOC contact) • T_AB_AND_N (for all three functions)
publicComments	**xsd:string**	O	None	Public comments.
justification	**xsd:string**	O	None	Justification.

Element name	Type	Opt/ Req	Default	Description
additionalInformation	**xsd:string**	O	None	Additional information.
otherCustomizedInfo	**xsd:string**	O	None	Spare field for user defined data to be included at the end of the report. The value of this field is included at the end of the report without any modification. This field should contain name value pair of user defined fields as they need to be included in the report.

INFO_REC

INFO_REC structure is used to specify Customer Information for Arin Reassign Simple report and Organization Information in other SWIP reports. The following table lists the required and optional attributes of INFO_REC structure for ARIN templates.

Table 2-69 INFO_REC attributes

Element name	Type	Opt/ Req	Default	Description
fullName	**xsd:string**	R	None	Full legal name of the customer receiving the reallocation or reassignment as it is registered with local, regional, or national authorities.
address	**xsd:string**	R	None	Street address of customer receiving the reallocation/reassignment.
city	**xsd:string**	R	None	City of the customer receiving the reallocation/reassignment.
stateOrProvince	**xsd:string**	O	None	Two-letter abbreviation of the state or province of the customer receiving the reallocation/reassignment
postalCode	**xsd:string**	O	None	Postal Code of the customer receiving the reallocation/reassignment

Element name	Type	Opt/ Req	Default	Description
countryCode	**xsd:string**	R	None	ISO-3166 two-character country code of the customer receiving the reallocation/reassignment. You can find a complete listing of country codes at: http://www.arin.net/community/countries.html
customizedInfo	**xsd:string**	O	None	Spare field for user defined data to be included after Country Code in Organization Information section in the report. The value of this field is included after Country Code field in Organization Information section without any modification. This field should contain name value pair of user defined fields as they need to be included in the report.

CONTACT_INFO_REC

The **CONTACT_INFO_REC** structure is used to specify Organization Contact Information and Network Technical Contact Information for the ARIN report. The following table lists the required and optional attributes of **CONTACT_INFO_REC** structure for ARIN templates.

Table 2-70 CONTACT_INFO_REC structure

Element name	Type	Opt/ Req	Default	Description
handle	**xsd:string**	R*	None	ARIN Org/Net POC Handle. *If POC Handle is provided, the rest of the contact attributes are not applicable and are ignored if provided.
contactType	**qipws:CONTACT_TYPE**	R	None	Org/Net POC Contact Type. Allowable values are: • P (for PERSON) • R (for ROLE)

Element name	Type	Opt/ Req	Default	Description
lastNameOrRole Account	**xsd:string**	R	None	Org/Net POC Last Name or Role Account. If the POC is a role account, enter the entire role account name, for example: “Network Operations Center” If the POC is a person, enter only the last name, for example: “Smith”.
firstName	**xsd:string**	O*	None	ORG POC First name. * Required if contactType=P. If contactType=R, then leave this field empty.
companyName	**xsd:string**	O*	None	Org/Net POC Company Name. * Required if contactType=R. This field is optional of contactType=P.
address	**xsd:string**	R	None	Org/Net POC Address.
city	**xsd:string**	R	None	Org/Net POC City.
stateOrProvince	**xsd:string**	O	None	Org/Net POC State/Province (two-letter abbreviation).
postalCode	**xsd:string**	O	None	Org/Net POC Postal Code.
countryCode	**xsd:string**	R	None	Org/Net POC Country Code. ISO-3166 two-character country code of the Org/Net POC. You can find a complete listing of country codes at: http://www.arin.net/community/countries.html
phoneNumber	**xsd:string**	R	None	Org/Net POC Office Phone Number
emailAddress	**xsd:string**	R	None	Org/Net POC E-mail Address
emailAddress2	**xsd:string**	O	None	Org/Net POC Alternate E-mail Address

Element name	Type	Opt/ Req	Default	Description
customizedInfo	**xsd:string**	O	None	Spare field for user-defined data to be included after E-mail Address information in the POC Contact section in the report. The value of this field is included after E-mail Address information in the POC Contact section without any modification. This field should contain name value pair of user-defined fields, as they need to be included in the report.

APNIC Reporting

Overview

VitalQIP contains optional support for registering address space with APNIC. APNIC maintains a database (WHOIS) of all registered IP address blocks and information about who is using this address space in the Asian Pacific.

If a seed pool is configured with APNIC support turned on, any block allocation that creates a VitalQIP subnet anywhere below the seed pool in the hierarchy will generate an inetnum report associated with the maintainer ID specified on the seed pool. VitalQIP does not generate inetnum reports for block allocation that are smaller than the minimum block size specified on the seed pool because APNIC does not wish to be notified about such block sizes.

For more information on the APNIC inetnum report, please access **www.apnic.net**.

Whenever an APNIC inetnum report is sent after a block has been allocated, VitalQIP stores the following as part of the block information:

- The type of APNIC inetnum report that was generated.
- The value of the network name field.

Whenever a Free Block operation is performed on a block that contains a valid inetnum report type and network name, a corresponding Free inetnum report is sent automatically.

VitalQIP does not place any restrictions on the field lengths or the values entered into the fields. In some cases, fields are not applicable if other fields have a value, and some fields may have length or content limitations but VitalQIP will not enforce any restriction. This is by design, since the format of the APNIC inetnum report is subject to change by APNIC. Please refer to **www.apnic.net** for more information on attributes in the inetnum report.

The following subsections contain details of the structures used to provide data for APNIC Inetnum reports.

APNIC_INETNUM_REPORT

This structure is used to specify data to be used for APNIC Inetnum Report during block allocation when seed pool is configured with APNIC support turned on. The following table lists the required and optional attributes of **`APNIC_INETNUM_REPORT`** structure.

Table 2-71 APNIC_INETNUM_REPORT structure

Element name	Type	Opt/ Req	Default	Description
networkNameOption	**qipws:RIR_NETWORK_NAME_OPTION_TYPE**	R	None	Specifies the option for defining **netName** element. Allowable values are: • **GENERATE_UNIQUE** – Use internally generated unique netName. • **USE_POOL_NAME** – Use pool name for netName • **USER_DEFINED** – Use user specified netName.
netName	**xsd:string**	O	None	Specifies the name of a range of IP address space. Required when networkNameOption is USER_DEFINED.
descr	**xsd:string**	R	None	A short description related to the object.
country	**xsd:string**	R	None	Two digit country code.
adminC	**qipws:INETNUM_ADMIN_CONTACT_TYPE**	R	None	Specifies how NIC handle for admin contact will be assigned. Allowable values are: • **AUTO-2** – User provides administrative contact information and APNIC database will assign the value automatically. • **NIC-Handle** – Use an existing NIC handle as an administrative contact.
adminNicHandler	**xsd:string**	O	None	NIC-handle of an on-site administrative contact. Required when adminC = NIC-handle.

Element name	Type	Opt/ Req	Default	Description
techC	**qipws:INETNUM_ TECH_CONTACT_TYPE**	R	None	Specifies how NIC handle for tech contact will be assigned. Allowable values are: • **AUTO-3** – User provides technical contact information and APNIC database will assign the value automatically. • **NIC-Handle** – Use an existing NIC handle as a technical contact.
techNicHandler	**xsd:string**	O	None	Nic-handle of technical contact. Required when techC = NIC-Handle
revSrv	**xsd:string**	O	None	DNS nameserver for a range of IP addresses represented by the inetnum or inet6num object that contans this attribute.
status	**qipws:APNIC_ INETNUM_STATUS_ TYPE**	R	None	Inetnum report status. Allowable values are: • ALLOCATED PORTABLE • ALLOCATED NON-PORTABLE • ASSIGNED PORTABLE • ASSIGNED NON-PORTABLE
remarks	**xsd:string**	O	None	Comments.
mntByPassword	**xsd:string**		None	mnt-by password.
mntLower	**xsd:string**	O	None	mnt-lower. Required when status is “ALLOCATED PORTABLE”
mntLowerPassword	**xsd:string**	O	None	mnt-lower password. Required when status is “ALLOCATED PORTABLE”

Element name	Type	Opt/ Req	Default	Description
mntRoutes	**xsd:string**	O	None	mnt-routes. This attribute references a **maintainer** object which is used in determining authorization for the creation of **route** objects.
notify	**xsd:string**	O		Specifies the e-mail address to which notifications of changes to an object should be sent.
changed	**xsd:string**	O	APNIC registry email address	<e-mail> [<date>] Specifies who submitted the update, and when the object was updated. The format of the date is YYYYMMDD; dates in the future are not allowed. If the date is not specified, the database will put the date when the update was actually processed.
source	**xsd:string**	O	APNIC	Specifies the registry where the object is registered. Should be "APNIC" for the APNIC Database.
customizedInfo	**xsd:string**	O	None	Spare field for user defined data to be included after source field in the report. Value of this field is included in the report without any modification. This field should contain name value pair of user defined fields as they need to be included in the report.
adminContactObject	**qipws:INETNUM_ CONTACT_OBJECT**	O	None	On-site Administrative Contact Information. Required when adminC = AUTO-2
techContactObject	**qipws:INETNUM_ CONTACT_OBJECT**	O	None	Technical Contact Information. Required when techC = AUTO-3

INETNUM_CONTACT_OBJECT

This structure is used to specify the information about administrative, technical or zone contact in inetnum report for RIPE, AfriNIC, or APNIC Registry. The schema for this structure is as shown below.

The following table lists the required and optional attributes of **INETNUM_CONTACT_OBJECT** structure.

Table 2-72 INETNUM_CONTACT_OBJECT structure

Element name	Type	Opt/ Req	Default	Description
person	**xsd:string**	R	None	Full name of the contact person.
address	**xsd:string**	R	None	Address of the contact.
phone	**xsd:string**	R	None	Phone number of the contact.
country	**xsd:string**	O	None	Country of the contact. This attribute is required for reports to APNIC registry. This attribute is not applicable for reports to RIPE/AfriNIC registry.
email	**xsd:string**	R	None	E-mail address of the contact.
remarks	**xsd:string**	O	None	Remarks.
notify	**xsd:string**	O	None	Email address to which notifications of changes to an object should be sent.
abuseMailbox	**xsd:string**	O	None	Email address to which abuse complaints should be sent. This attribute is not applicable for reports to AfriNIC registry.
mntBy	**xsd:string**	O	None	Identifier of a registered mntner object used for authorisation of operations performed on the object that contains this attribute.
changed	**xsd:string**	O	Registry e-mail address	<e-mail>[<date>]
source	**xsd:string**	O	APNIC	Specifies the registry. Should be “APNIC” for the APNIC database “RIPE” for the RIPE database, and “AFRINIC” for AfriNIC registry.

Element name	Type	Opt/ Req	Default	Description
customizedInfo	**xsd:string**	O	None	Spare field for user defined data to be included at the end of contact section in the report. The value of this field is included at the end of contact section without any modification. This field should contain name value pair of user defined fields as they need to be included in the report.

RIPE Reporting

VitalQIP contains optional support for registering address space with RIPE. RIPE maintains a database (WHOIS) of all registered IP address blocks and information about who is using this address space in Europe and the Middle-East.

If a seed pool is configured with RIPE support turned on, any block allocation that creates a VitalQIP subnet anywhere below the seed pool in the hierarchy will generate an inetnum report associated with a maintainer ID specified on the seed pool. To generate an inetnum report, user must specify either *ripeReport* element. The specified information will be formatted into the selected inetnum report and emailed to the RIPE Email address specified on the seed pool.

VitalQIP does not generate inetnum reports for block allocations that are smaller than the minimum block size specified on the seed pool because RIPE does not wish to be notified about such block sizes.

For information the RIPE inetnum report, please access **www.ripe.net**.

Whenever a RIPE inetnum report is sent after a block has been allocated, VitalQIP stores the following as part of the block information:

- The type of RIPE inetnum report that was generated.
- The value of the network name field.

Whenever a Free Block operation is performed on a block that contains a valid inetnum report type and network name, a corresponding Free inetnum report is sent automatically.

VitalQIP does not place any restrictions on the field lengths or the values entered into the fields. In some cases, some fields are not applicable if other fields have a value and some fields may have length or content limitations but VitalQIP will not enforce any restriction. This is by design since the format of the RIPE inetnum report is subject to change by RIPE. Please refer to **www.ripe.net** for more information on attributes in the inetnum report.

The following subsections contain details of the structures used to provide data for RIPE Inetnum Reports.

RIPE_INETNUM_REPORT

This structure is used to specify data to be used for RIPE Inetnum Report during block allocation when seed pool is configured with RIPE support turned on. Following table lists the required and optional attributes of **`RIPE_INETNUM_REPORT`** structure.

Table 2-73 RIPE_INETNUM_REPORT structure

Element name	Type	Opt/ Req	Default	Description
networkNameOption	**qipws:RIR_NETWORK_ NAME_OPTION_TYPE**	R	None	Specifies the option for defining netName element. Allowable values are: • **GENERATE_UNIQUE** – Use internally generated unique netName. • **USE_POOL_NAME** – Use pool name for netName • **USER_DEFINED** – Use user specified netName.
descr	**xsd:string**	R	None	A short description related to the object.
country	**xsd:string**	R	None	Two digit country code.
status	**APNIC_INETNUM_ STATUS_TYPE**	R	None	Inetnum report status. Allowable values for IPv4 address blocks are: • ALLOCATED PA • ALLOCATED PI • ALLOCATED UNASSIGNED • ASSIGNED PA • ALLOCATED PA Allowable values for IPv6 address blocks are: • ALLOCATED-BY-LIR • ASSIGNED

Element name	Type	Opt/ Req	Default	Description
adminC	**qipws:INETNUM_ADMIN_CONTACT_TYPE**	R	None	Specifies how NIC handle for on-site administrative contact will be assigned. Allowable values are: • **AUTO-2** – User provides administrative contact information and Ripe database will assign the value automatically. • **NIC-Handle** – Use an existing NIC handle as an administrative contact.
techC	**qipws:INETNUM_TECH_CONTACT_TYPE**	R	None	Specifies how NIC handle for tech contact will be assigned. Allowable values are: • **AUTO-3** – User provides technical contact information and Ripe database will assign the value automatically. • **NIC-Handle** – Use an existing NIC handle as a technical contact.

Element name	Type	Opt/ Req	Default	Description
org	**qipws:RIPE_ORG_ CONTACT_TYPE**	O	None	Specifies how NIC handle for organization contact will be assigned. Required when status is one of following: • ALLOCATED PA • ALLOCATED PI • ALLOCATED UNSPECIFIED • ALLOCATED-BY-LIR This element is ignored for all other inetnum status. Allowable values are: • **AUTO-1** – User provides organization contact information and Ripe database will assign the value automatically. • **NIC-Handle** – Use an existing NIC handle as an organization contact.
netName	**xsd:string**	O	None	Specifies the name of a range of IP address space. Required when **networkNameOption** is USER_DEFINED.
orgNicHandler	**xsd:string**	O	None	NIC-handle of an on-site organization contact. Required when org = NIC-handle.
adminNicHandler	**xsd:string**	O	None	NIC-handle of an on-site administrative contact. Required when adminC = NIC-handle.
techNicHandler	**xsd:string**	O	None	Nic-handle of technical contact. Required when techC = NIC-Handle.
revSrv	**xsd:string**	O	None	DNS nameserver for a range of IP addresses represented by the inetnum or inet6num object that contans this attribute.

Element name	Type	Opt/ Req	Default	Description
remarks	**xsd:string**	O	None	Remarks.
mntByPassword	**xsd:string**	O	Pool Main-tainer password	mnt-by password.
mntLower	**xsd:string**	O	None	mnt-lower.
mntLowerPassword	**xsd:string**	O	None	mnt-lower password.
mntDomains	**xsd:string**	O	None	Identifier of a registered mntner object used for reverse domain authorization.
mntRoutes	**xsd:string**	O	None	mnt-routes. This attribute references a **maintainer** object which is used in determining authorisation for the creation of **route** objects.
mntIrt	**xsd:string**	O	None	mnt-irt. This attribute references an irt object.
notify	**xsd:string**	O	None	Specifies the e-mail address to which notifications of changes to an object should be sent.
changed	**xsd:string**	O	RIPE registry e-mail address	<e-mail> [<date>] Specifies who submitted the update, and when the object was updated. The format of the date is YYYYMMDD; dates in the future are not allowed. If the date is not specified, the database will put the date when the update was actually processed.
source	**xsd:string**	O	RIPE	Specifies the registry where the object is registered. Should be “RIPE” for the RIPE database.

Element name	Type	Opt/ Req	Default	Description
customizedInfo	**xsd:string**	O	None	Spare field for user defined data to be included after source info in the report. The value of this field is included after the source field in the report without any modification. This field should contain name value pair of user-defined fields, as they need to be included in the report.
organisation ContactObject	**qipws:RIPE_ORG_ OBJECT**	O	None	Organization Contact Information. Required when org = AUTO-1.
adminContact Object	**qipws:INETNUM_ CONTACT_OBJECT**		None	On-site Administrative Contact Information. Required when adminC = AUTO-2.
techContactObject	**qipws:INETNUM_ CONTACT_OBJECT**		None	Technical Contact Information. Required when techC = AUTO-3.

RIPE_ORG_OBJECT

This structure is used to specify the information about the organization object in an inetnum report for the RIPE Registry. The schema for this structure is as shown in the following table. It lists the required and optional attributes of **RIPE_ORG_OBJECT** structure.

Table 2-74 RIPE_ORG_OBJECT structure

Element name	Type	Opt/ Req	Default	Description
orgName	**xsd:string**	R	None	Organization Name.
orgType	**qipws:INETNUM_ORG_ TYPE**	R	None	Allowable values are: • IANA • LIR • NIR • NON-REGISTRY • RIR

Element name	Type	Opt/ Req	Default	Description
address	**xsd:string**	R	None	Address of the contact.
email	**xsd:string**	R	None	E-mail address of the contact.
mntRef	**xsd:string**	R	None	mnt-ref.
descr	**xsd:string**	O	None	A short description related to the organization.
remarks	**xsd:string**	O	None	Remarks.
phone	**xsd:string**	O	None	Phone number of the contact.
faxNo	**xsd:string**	O	None	Fax number of the contact.
refNfy	**xsd:string**	O	None	ref-nfy.
notify	**xsd:string**	O	None	Specifies the e-mail address to which notifications of changes to this organization object should be sent.
abuseMailbox	**xsd:string**	O	None	Email address to which abuse complaints should be sent.
changed	**xsd:string**	O	RIPE Registry e-mail	<e-mail>[<date>] Specifies who submitted the update, and when the object was updated.
source	**xsd:string**	O	RIPE	Specifies the registry. Should be “RIPE” for the RIPE database.
customizedInfo	**xsd:string**	O	None	Spare field for user-defined data to be included after source field in the organization section of the report. The value of this field is included after the source field in the organization section without any modification. This field should contain name value pair of user-defined fields, as they need to be included in the report.

AfriNIC reporting

Overview

VitalQIP contains optional support for registering address space with AfriNIC. AfriNIC maintains a database (WHOIS) of all registered IP address blocks, and information about who is using this address space in Africa.

If a seed pool is configured with AfriNIC support turned on, any block allocation that creates a VitalQIP subnet anywhere below the seed pool in the hierarchy will generate an inetnum report associated with a maintainer ID specified on the seed pool. To generate an inetnum report, user must specify either ripeReport element. The specified information will be formatted into the selected inetnum report and emailed to the AfriNIC Email address specified on the seed pool.

VitalQIP does not generate inetnum reports for block allocations that are smaller than the minimum block size specified on the seed pool because AfriNIC does not wish to be notified about such block sizes.

For information on the AfriNIC inetnum report, please access **www.afrinic.org**.

Whenever a AfriNIC inetnum report is sent after a block has been allocated, VitalQIP stores the following as part of the block information:

- The type of AfriNIC inetnum report that was generated.
- The value of the network name field.

Whenever a Free Block operation is performed on a block that contains a valid inetnum report type and network name, a corresponding Free inetnum report is sent automatically.

VitalQIP does not place any restrictions on the field lengths or the values entered into the fields. In some cases, some fields are not applicable if other fields have a value, and some fields may have length or content limitations but VitalQIP will not enforce any restriction. This is by design since the format of the AfriNIC inetnum report is subject to change by AfriNIC. Please refer to **www.afrinic.org** for more information on attributes in the inetnum report.

The following subsections contain details of the structures used to provide data for AfriNIC Inetnum Reports.

AFRINIC_INETNUM_REPORT

This structure is used to specify data to be used for AfriNIC Inetnum Report during block allocation when the seed pool is configured with AfriNIC support turned on. The following table lists the required and optional attributes of the **`AFRINIC_INETNUM_REPORT`** structure.

Table 2-75 AFRINIC_INETNUM_REPORT structure

Element name	Type	Opt/ Req	Default	Description
networkNameOption	**qipws:RIR_NETWORK_NAME_OPTION_TYPE**	R	None	Specifies the option for defining netName element. Allowable values are: • **GENERATE_UNIQUE** – Use internally generated unique netName. • **USE_POOL_NAME** – Use pool name for netName. • **USER_DEFINED** – Use user specified netName.
descr	**xsd:string**	R	None	A short description related to the object.
country	**xsd:string**	R	None	Two digit country code.
status	**qipws:AFRINIC_INETNUM_STATUS_TYPE**	R	None	Inetnum report status. Allowable values for IPv4 address blocks are: • ALLOCATED PA • ALLOCATED PI • ASSIGNED PA • ASSIGNED PI • EARLY-REGISTRATION • NOT-SET • SUB-ALLOCATED PA • UNSPECIFIED Allowable values for IPv6 address blocks are: • ALLOCATED-BY-RIR • ASSIGNED PA • ASSIGNED PI

Element name	Type	Opt/ Req	Default	Description
adminC	**qipws:INETNUM_ ADMIN_CONTACT_ TYPE**	R	None	Specifies how NIC handle for on-site administrative contact will be assigned. Allowable values are: • **AUTO-2** – User provides administrative contact information and AfriNIC database will assign the value automatically. • **NIC-Handle** – Use an existing NIC handle as an administrative contact.
techC	**qipws:INETNUM_ TECH_CONTACT_TYPE**	R	None	Specifies how NIC handle for tech contact will be assigned. Allowable values are: • **AUTO-3** – User provides technical contact information and AfriNIC database will assign the value automatically. • **NIC-Handle** – Use an existing NIC handle as a technical contact.
org	**qipws:AFRINIC_ORG _CONTACT_TYPE**	O	None	Specifies how NIC handle for organization contact will be assigned. Required when status is one of following: • ALLOCATED PA • ALLOCATED PI • UNSPECIFIED This element is ignored for all other inetnum status. Allowable values are: • **AUTO-1** – User provides organization contact information and AfriNIC database will assign the value automatically. • **NIC-Handle** – Use an existing NIC handle as an organization contact.

Element name	Type	Opt/ Req	Default	Description
netName	**xsd:string**	O	None	Specifies the name of a range of IP address space. Required when networkNameOption is USER_DEFINED.
orgNicHandler	**xsd:string**	O	None	NIC-handle of an on-site organization contact. Required when **org = NIC-Handle**.
techNicHandler	**xsd:string**	O	None	NIC-handle of technical contact. Required when **techC = NIC-Handle**.
revSrv	**xsd:string**	O	None	DNS nameserver for a range of IP addresses represented by the inetnum or inet6num object that contans this attribute.
remarks	**xsd:string**	O	None	Remarks.
mntByPassword	**xsd:string**	O	Pool Maintainer password	mnt-by password.
mntLower	**xsd:string**	O	None	mnt-lower.
mntLowerPassword	**xsd:string**	O	None	mnt-lower password.
mntDomains	**xsd:string**	O	None	Identifier of a registered maintainer object used for reverse domain authorization.
mntRoutes	**xsd:string**	O	None	mnt-routes. This attribute references a maintainer object that is used in determining authorization for the creation of route objects.
mntIrt	**xsd:string**	O	None	This attribute references an irt object.
notify	**xsd:string**	O	None	Specifies the e-mail address to which notifications of changes to an object should be sent.

Element name	Type	Opt/ Req	Default	Description
changed	**xsd:string**	O	AfriNIC registry email address	<e-mail> [<date>] Specifies who submitted the update, and when the object was updated. The format of the date is YYYYMMDD; dates in the future are not allowed. If the date is not specified, the database will put the date when the update was actually processed.
source	**xsd:string**	O	AFRINIC	Specifies the registry where the object is registered. Should be "AFRINIC" for the AfriNIC Database.
customizedInfo	**xsd:string**	O	None	Spare field for user-defined data to be included after the source information in the report. The value of this field is included after the source field in the report without any modification. This field should contain name value pair of user-defined fields as they need to be included in the report.
organisation ContactObject	**qipws:AFRINIC_ORG_OBJECT**	O	None	Organization Contact Information. Required when **org = AUTO-1**.
adminContact Object	**qipws:INETNUM_ CONTACT_OBJECT**		None	On-site Administrative Contact Information. Required when **adminC = AUTO-2**.
techContactObject	**qipws:INETNUM_ CONTACT_OBJECT**		None	Technical Contact Information. Required when **techC = AUTO-3**.

AFRINIC_ORG_OBJECT

This structure is used to specify the information about organization object in inetnum report for AfriNIC Registry. The schema for this structure is as shown below.

The following table lists the required and optional attributes of **ORG_CONTACT_OBJECT** structure.

Table 2-76 ORG_CONTACT_OBJECT structure

Element name	Type	Opt/ Req	Default	Description
orgName	**xsd:string**	R	None	Organization Name.
orgType	**qipws:INETNUM_ORG_ TYPE**	R	None	Allowable values are: • IANA • LIR • NIR • NON-REGISTRY • RIR
address	**xsd:string**	R	None	Address of the contact.
email	**xsd:string**	R	None	E-mail address of the contact.
mntRef	**xsd:string**	R	None	mnt-ref.
descr	**xsd:string**	O	None	A short description related to the organization.
remarks	**xsd:string**	O	None	Remarks.
phone	**xsd:string**	O	None	Phone number of the contact.
faxNo	**xsd:string**	O	None	Fax number of the contact.
country	**xsd:string**	O	None	Two digit country code.
org	**xsd:string**	O	None	Organization.
refNfy	**xsd:string**	O	None	ref-nfy.
notify	**xsd:string**	O	None	Specifies the e-mail address to which notifications of changes to this organization object should be sent.
abuseMailbox	**xsd:string**	O	None	Email address to which abuse complaints should be sent.
changed	**xsd:string**	O	AFRINIC Registry e-mail	<e-mail>[<date>] Specifies who submitted the update, and when the object was updated.
source	**xsd:string**	O	AFRINIC	Specifies the registry. Should be “AFRINIC” for the AfriNIC database.

Element name	Type	Opt/ Req	Default	Description
customizedInfo	**xsd:string**	O	None	Spare field for user-defined data to be included after source field in the organization section of the report. The value of this field is included after the source field in the organization section without any modification. This field should contain name value pair of user-defined fields, as they need to be included in the report.

Address templates

Overview

Purpose

Address templates dictate types and number of objects that are assigned when a subnet is created during block allocation. The following operations are supported:

- Add
- Associate
- CountAssociations
- Copy
- Delete
- Dissociate
- FindAssociations
- Get
- Search
- Update

Details of the messages for each operation is given in the following table.

Table 2-77 Message details

Action	Operation	Incoming message	Outgoing message
Associate	VQIPManager_Associate Request	AssociateRequest	Response
Add	VQIPManager_AddRequest	AddRequest	Response
Copy	VQIPManager_Copy Request	CopyRequest	Response
CountAssociations	VQIPManager_Count Associations	CountAssociations Request	CountAssociations Response
Delete	VQIPManager_DeleteRequest	DeleteRequest	Response
Dissociate	VQIPManager_Dissociate Request	DissociateRequest	Response
FindAssociations	VQIPManager_Find Associations	FindAssociationsRequest	FindAssociations Response
Get	VQIPManager_GetRequest	GetRequest	GetResponse

Action	Operation	Incoming message	Outgoing message
Search	VQIPManager_SearchRequest	SearchRequest	SearchResponse
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

Add, delete, get, search and update operations

The northbound system can add address templates using **VQIPManager_AddRequest** operation using **AddRequest** message. **QIPManager_DeleteRequest** operation should be used for deleting an address template. **VQIPManager_GetRequest** operation should be used for retrieving a single address template and **VQIPManager_SearchRequest** operation should be used for retrieving a list of address templates. Updates can be done by invoking the **VQIPManager_UpdateRequest** operation, which uses the **UpdateRquest** message described Table 2-24, "UpdateRequest structure" (p. 2-30).

ADDR_TEMPL_REC object is the input object for Address Template operations. Details of attributes required for each operation are included in the following tables.

Table 2-78 ADDR_TEMPL_REC attributes

name

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	R	R	O	R
Description:	Unique name of up to 64 alphanumeric characters.				

addressType

Type:	**qipws:V4V6_TYPE**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	N/A	O	R	R
Description:	Determines type to be used during block allocation.				

description

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Additional information.				

v4ManagedRanges

Type:	**qipws:ARRAY_OF_V4MANAGED_RANGE_REC**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Optional list of V4 managed ranges.				

v6ManagedRanges

Type:	**qipws:ARRAY_OF_V6MANAGED_RANGE_REC**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Optional list of V6 managed ranges.				

ARRAY_OF_V4MANAGED_RANGE_REC

Address templates that are of **addressType** V4 can have zero or more associated objects of type **V4MANAGED_RANGE_REC**, as described in the following table.

Table 2-79 v4ManagedRanges structure

Part name	Type	Opt/ Req	Default	Description
v4Manages Ranges	**qipws:V4MANAGED_RANGE_REC**	O	None	An array of one or more **V4MANAGED_RANGE_REC** objects. **V4MANAGED_RANGE_REC** attributes are described in Table 2-80, "V4MANAGED_RANGE_REC attributes" (p. 2-133).

Table 2-80 V4MANAGED_RANGE_REC attributes

addressType

Type:	**qipws:ADDRESS_TYPE**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	N/A	O	O	R
Description:	Determines type to be used during block allocation. This is an enumerated type. Refer to "ADDRESS_TYPE" (p. 3-6) for definition of **ADDRESS_TYPE**.				

objectClass

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	N/A	N/A	N/A	R
Description:	Type of object such as WorkStation, PC, printer, and so on. Valid values should be determined by using the **VQIP_Manager_GetAll** operation.				

domain

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	N/A	N/A	N/A	R
Description:	Valid values should be determined by using the **VQIP_Manager_GetAll** operation.				

dhcpServer

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R for Dynamic addressType	N/A	N/A	N/A	R for Dynamic addressType
Description:	Valid values should be determined by using the **VQIP_Manager_GetAll** operation.				

dhcpOptTemp

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R for Dynamic addressType	N/A	N/A	N/A	R for dynamic
Description:	DHCP Option Template to be associated with this scope of addresses. Valid values should be determined by using the **VQIP_Manager_GetAll** operation.				

dhcpScopePcy

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	DHCP Scope Policy Template to be associated with this scope of addresses. Valid values should be determined by using the **VQIP_Manager_GetAll** operation.				

leaseTime

Type:	**long**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Lease time of dynamic objects.				

userClass

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Only used when **addressType** is dynamic. Valid values should be determined by using the **VQIP_Manager_GetAll** operation.				

vendorClass

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Only used when **addressType** is dynamic. Valid values should be determined by using the **VQIP_Manager_GetAll** operation.				

startOffset

Type:	**long**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R for Dynamic addressType	N/A	N/A	N/A	O
Description:					

endOffset

Type:	**long**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R for Dynamic addressType	N/A	N/A	N/A	O
Description:					

numberOfObjects

Type:	**long**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R for Dynamic addressType	N/A	N/A	N/A	O
Description:					

ARRAY_OF_V6MANAGED_RANGE_REC

Address Templates that are of **addressType** V6 can have zero or more associated objects of type **V6MANAGED_RANGE_REC**, as described in the following table.

Table 2-81 V6MANAGED_RANGE_REC structure

Part name	Type	Opt/ Req	Default	Description
V6ManagesRanges	**qipws:V6MANAGED_ RANGE_REC**	O	None	An array of one or more **V6MANAGED_RANGE_REC** objects. **V4MANAGED_RANGE_REC** attributes are described in Table 2-82.

Table 2-82 V6MANAGED_RANGE_REC attributes

addressType

Type:	**qipws:ADDRESS_TYPE**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	N/A	N/A	N/A	R
Description:	Determines type to be used during block allocation.				

domain

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	N/A	N/A	N/A	R
Description:	Valid values should be determined by using the **VQIP_Manager_GetAll** operation.				

interface

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	The interface name (on each of the nodes), to which the IP Address will be associated with. N/A for IPv4.				

nodeType

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	N/A	N/A	N/A	R
Description:	Valid values should be determined by using the **VQIP_Manager_GetAll** operation.				

startOffset

Type:	**long**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Either **startOffset** or **endOffset** must of populated.				

endOffset

Type:	**long**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Required if **startOffset** is not populated.				

numberOfObjects

Type:	**long**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Required if **startOffset** is not populated.				

AssociateRequest Message

The **VQIPManager_AssociateRequest** operation should be used for associating an address template with a rule. Only rules of "Used" type can be associated with an address template. Additionally, IPv4 rules can only be associated with IPv4 templates and IPv6 rules can only be associated with IPv6 Address templates. This operation uses **AssociateRequest** message. Details of **AssociateRequest** message are given in Table 2-6, "AssociateRequest details" (p. 11). **srcKey** is of type **RULE_KEYS** and **targetKey** is of type **ADDR_TEMPL_KEYS**.

CopyRequest message

The **VQIPManager_CopyRequest** operation should be used for copying an address template. This operation uses **CopyRequest** message. The details of **CopyRequest** message are given in "CopyRequest operation" (p. 2-222). The **from** parameters are **ADDR_TEMPL_KEYS** that identify the address template being copied. The **to** parameter is a string that defines the name of destination address template.

CountAssociations message

The **VQIPManager_CountAssocationsRequest** operation should be used for counting the number of rules associated with an address template. **objectKey** is of type **ADDR_TEMPL_KEYS**.

DissociateRequest message

The **VQIPManager_DissociateRequest** operation should be used for dissociating an address template with a rule. Details of **DissociateRequest** message are given in Table 2-12, "DissociateRequest details" (p. 18). **srcKey** is of type **RULE_KEYS** and **targetKey** is of type **ADDR_TEMPL_KEYS**.

FindAssociationsRequest message

The **VQIPManager_FindAssociationsRequest** operation should be used for retrieving the list of rules associated with an address template. Details of **FindAssociationsRequest** message are given in Table 2-14, "FindAssociations

details" (p. 20). **objectKey** is of type **ADDR_TEMPL_KEYS**. If the address template identified by **objectKey** has any associated rules, objects of type **RULE_KEYS** are returned.

Address range

Overview

Purpose

An “address range” is a container that can contain one or more IPv6 Subnets. It is defined by its prefix and length, and provides a shortcut to allow administrators to organize IPv6 subnets. An address range is optional in VitalQIP.

The following operations are supported on address ranges:

- Add
- Delete
- Get
- Update

The **`ADDRESS_RANGE_REC`** structure is used for operations on address ranges.

Required messages for each operations are given the following table.

Table 2-83 ADDRESS_RANGE_REC message attibutes

Action	Operation	Incoming message	Outgoing message
Add	VQIPManager_AddRequest	AddRequest	Response
Delete	VQIPManager_DeleteRequest	DeleteRequest	Response
Get	VQIPManager_GetRequest	GetRequest	GetResponse
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

The following table lists the required and optional attributes for operations on address range records.

Table 2-84 ADDRESS_RANGE_REC attributes

startAddress

Type:	**`xsd:string`**			
Operation:	Add	Delete	Get	Search
Optional/Required:	R	R	R	R
Description:	IPv6 address.			

prefixLength

Type:	**xsd:short**			
Operation:	Add	Delete	Get	Search
Optional/Required:	R	R	R	R
Description:	IPv6 address prefix length.			

name

Type:	**xsd:string**			
Operation:	Add	Delete	Get	Search
Optional/Required:	R	N/A	N/A	R
Description:	Name or an unique identifier for the address range.			

parentBlockAddr

Type:	**xsd:string**			
Operation:	Add	Delete	Get	Search
Optional/Required:	R	N/A	N/A	R
Description:	IPv6 address of parent address block.			

Add address range

Overview

The northbound system invokes VQIPManager_AddRequest operation with AddRequest message containing **`ADDRESS_RANGE_REC`** structure for adding address range. The **`ADDRESS_RANGE_REC`** structure described in Table 2-84, “ADDRESS_RANGE_REC attributes” (p. 140) is used for operations on address range records.

Delete address range

Overview

The northbound system invokes **`VQIPManager_DeleteRequest`** operation with DeleteRequest message containing ADDRESS_RANGE_REC structure for deleting address range. The **`ADDRESS_RANGE_REC`** structure described in Table 2-84, “ADDRESS_RANGE_REC attributes” (p. 140) is used for operations on address range records.

Get address range

Overview

The northbound system invokes **`VQIPManager_GetRequest`** operation with GetRequest message containing **`ADDRESS_RANGE_REC`** structure for retrieving attributes of a specific address range. If the specified address range is found, attributes of that address range are returned. If the address range is not found, an empty record is returned in the **`Response`** message.

Update Address Range

Overview

The northbound system invokes **`VQIPManager_UpdateRequest`** operation with UpdateRequest containing **`ADDRESS_RANGE_REC`** structure for updating an address range. The **`ADDRESS_RANGE_REC`** structure described in Table 2-84, “ADDRESS_RANGE_REC attributes” (p. 140).

For an Update operation, all data fields for a record, those that are being updated as well as those that are not being updated, should be sent in as input. Fields for which values are being nulled are sent in as the attribute names with no values associated with them.

Clone server

Overview

Purpose

Cloning of DNS and DHCP servers is supported. The message required for cloning both DNS and DHCP servers is given in the following table.

Table 2-85 Clone server operations and messages

Action	Operation	Incoming message	Outgoing message
Copy	VQIPManager_CopyRequest	CopyRequest	Response

Please refer to "Request structure - CopyRequest message" (p. 2-13) for explanation of attributes of CopyRequest message. The from and to attributes for CopyRequest are of type SERVER_KEY.

The **`addlParams`** attribute is relevant only to cloning of DNS Servers. For cloning of DNS servers, two additional parameters are relevant.

Table 2-86 DNS server parameters

Name	Value	Description
`includeZones`	true/false	include zone data from the source server in the destination server. Default is true.
`makeSecondary`	true/false	Include zone data from the source server in the destination server and make destination server secondary to all zones. Default is false.

Refer to "Request for Clone DNS Server" (p. 4-21) for an example of a clone DNS server request.

DNS - ACL templates

Overview

Purpose

An Access Control List (ACL) data type is used in several DNS operations, including the definition of an ACL template. Each access control entry in the list has positional significance. Each entry can be one of several different types (for example it may define an IPv4 network, or an IPv6 address). Therefore, the definition of an entry provides a choice of datatypes, but only one may be used at a time. The datatype's value may be negated by prepending a "!". For example, to reference all IPv4 networks, except "192.168.1.0/24", use a value of "!192.168.1.0/24".

The following table documents the operations, and messages, available for managing DNS ACL templates. Note that that the delete operation provides an alternate form with the ability to force a template delete, even if it is currently referenced elsewhere

Table 2-87 DNS - ACL templates operations and messages

Action	Operation	Incoming message	Outgoing message
Add	VQIPManager_AddRequest	AddRequest	Response
Copy	VQIPManager_CopyRequest	CopyRequest	Response
Delete	VQIPManager_DeleteRequest	DeleteRequest	Response
DeleteForcedRequest	VQIPManager_DeleteForced Request	DeleteForcedRequest	Response
Get	VQIPManager_GetRequest	GetRequest	GetResponse
PageSearch	VQIPManager_PageSearch Request	PageSearchRequest	PageSearchResponse
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

DNS - ACL template attributes

DNS_ACL_TEMPLATE_REC

The following table describes the definitions for **DNS_ACL_TEMPLATE_REC**.

Table 2-88 DNS_ACL_TEMPLATE_REC attributes

name

Type:	**xsd:string**	
Operation:	Get	Update
Optional/ Required:	R	R
Description:	DNS ACL Template's name.	

acl

Type:	**qipws:DNS_ACL_TYPE**	
Operation:	Get	Update
Optional/ Required:	O	O
Description:	Holds a list of entries. An empty list is allowed.	

An **acl** contains a **DNS_ACL_TYPE** list. Though definition of **DNS_ACL_TYPE** contains several elements, the schema defines it as a <choice>. This means only one of the elements may be populated in each instance of **DNS_ACL_TYPE**. Furthermore, WSDL compilers may generate additional code to support <choice>. AXIS will layer the choices within an extra DNS_ACL_TYPE*Choice* object whereas gSoap will generate a union.

Table 2-89 DNS_ACL_TYPE choices

Choice Name	Type	Description
ipv4_address	**xsd:string**	An IPv4 address, such as 192.168.1.2.
ipv4_network	**xsd:string**	An IPv4 network, such as 192.168.1.0/24.
ipv6_address	**xsd:string**	An IPv6 address.
ipv6_network	**xsd:string**	An IPv6 network.
template	**xsd:string**	The name of another ACL template.
key	**xsd:string**	A DNS ACL key.

Choice Name	Type	Description
other	**qipws:DNS_ACT_OTHER_TYPE**	One of these predefined keywords: ANY, NONE, LOCALHOST or LOCALNETS. To negate its value, a "!" may be prepended before LOCALHOST and LOCALNETS.
text	**xsd:string**	Free format text subject to DNS server syntax constraints.

ARRAY_OF_DNS_OPTIONS structure

The following table documents the **qipws:ARRAY_OF_DNS_OPTIONS** structure, used by several DNS messages types, to represent a collection of DNS options.

Table 2-90 qipws:ARRAY_OF_DNS_OPTIONS attributes

Part Name	Type	Opt/Req	Description
option	**qipws:DNS_OPTION_TYPE**	O	List of 0 or more DNS Options.

An option contains a **DNS_OPTION_TYPE** list. Though definition of **DNS_OPTION_TYPE** contains both **value** and **AddrMatchValue** elements, the schema defines that pair as a <choice>. This means only one of the two may be populated in each instance of **DNS_OPTION_TYPE**. Furthermore, WSDL compilers may generate additional code to support <choice>. For example, AXIS will layer the choices within an extra DNS_ACL_TYPE*Choice* object whereas gSoap will generate a union.

Table 2-91 DNS_OPTION_TYPE attributes

Part Name	Type	Opt/Req	Description
name	**xsd:string**	R	Option's name; constrained to the predefined list of options in *dns_zone_options.xml*.

Part Name	Type	Opt/Req	Description
value	**xsd:string**	R (pick one)	For use when the DNS option's value is a string. The appropriate syntax depends on the specific DNS option.
AddrMatchValue	**qipws:DNS_ACL_TYPE**		For use when a DNS option's value is an address match list. Refer to Table 2-89, "DNS_ACL_TYPE choices" (p. 148) for more information.

DNS Views

Overview

Purpose

These are the operations for administering DNS Views. Each DNS View has a sequential position in relation to the other views. When a new view is added, it is automatically appended to the end of the list. All views may be efficiently listed using **`GetAllRequest`** and the results are sorted by their order number. A **`Sequence`** request may be used to change the ordering.

Table 2-92 DNS Views operations and messages

Action	Operation	Incoming message	Outgoing message
Add	VQIPManager_AddRequest	AddRequest	Response
Delete	VQIPManager_DeleteRequest	DeleteRequest	Response
Get	VQIPManager_GetRequest	GetRequest	GetResponse
List	VQIPManager_GetAllRequest	GetAllRequest	GetAllResponse
PageSearch	VQIPManager_PageSearchRequest	PageSearchRequest	PageSearchResponse
Sequence	VQIPManager_SequenceRequest	SequenceRequest	Response
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

DNS - View attributes

DNS_VIEW_REC

`ReqObject` contains an instance of **`DNS_VIEW_REC`** for the actions Add, Delete, Get, and Update. The following table lists its required and optional attributes.

Table 2-93 DNS_VIEW_REC attributes

name

Type:	**`xsd:string`**			
Operation:	Get	Add	Delete	Update
Optional/ Required:	R	R	R	R
Description:	DNS View's name.			

matchClients

Type:	**`qipws:DNS_ACL_TYPE`**			
Operation:	Get	Add	Delete	Update
Optional/ Required:	N/A	O	N/A	O
Description:	Uses an Access Control List. Refer to Table 2-89, "DNS_ACL_TYPE choices" (p. 148) for more information.			

matchDestinations

Type:	**`qipws:DNS_ACL_TYPE`**			
Operation:	Get	Add	Delete	Update
Optional/ Required:	N/A	O	N/A	O
Description:	Uses an Access Control List. Refer to Table 2-89, "DNS_ACL_TYPE choices" (p. 148) for more information.			

matchRecursiveOnly

Type:	**`xsd:boolean`**			
Operation:	Get	Add	Delete	Update
Optional/ Required:	N/A	O	N/A	O
Description:	Refer to the RFCs for details.			

dnsOptionList

Type:	**qipws:ARRAY_OF_DNS_OPTIONS**			
Operation:	Get	Add	Delete	Update
Optional/ Required:	N/A	O	N/A	O
Description:	An ordered list of options. Refer to Table 2-91, "DNS_OPTION_TYPE attributes" (p. 149) for more information.			

optionalAttributeList

Type:	**qipws:UdaList**			
Operation:	Get	Add	Delete	Update
Optional/ Required:	N/A	O	N/A	O
Description:	List of user defined attributes.			

The following table lists the attribute for the DNS View action **SequenceRequest**.

Table 2-94 SequenceRequest attribute

Part Name	Type	Opt/Req	Description
reqObject	**qipws:DNS_VIEW_KEY**	R	An ordered list of DNS View keys.

DNS_VIEW_KEY

ReqObject contains a list of **DNS_VIEW_KEYs** for the **Sequence** action.

Table 2-95 DNS_VIEW_KEY attributes

Part Name	Type	Opt/Req	Description
name	**xsd:string**	R	View's name.

SEARCH_DNS_VIEW_REC

For a **PageSearch** action, the reqObject takes a **qipws:SEARCH_DNS_VIEW_REC** containing these required and optional attributes.

Table 2-96 SEARCH_DNS_VIEW_REC attributes

Part Name	Type	Opt/Req	Description
name	**xsd:string**	O	View's name. Leaving it unset is equivqalent to "*".

Part Name	Type	Opt/Req	Description
optionalAttributeList	**qipws:UdaList**	O	List of zero, or more, user defined attributes.

DNS View - zones

Overview

Purpose

This section defines a group of operations for adding a DNS Zone to a view. A zone may either be created specifically for a DNS View, or it may be an existing global zone that has been shared with a view. Different messages, and capabilities, are provided to handle both scenarios. Add and Delete actions apply to view specific zones, whereas Share and Unshare are for global zones. Get, Update, and Search actions support both zone types, but their message contents differ by type.

Table 2-97 DNS View - zones operations and messages

Action	Operation	Incoming message	Outgoing message
Add	VQIPManager_AddRequest	AddRequest	Response
Delete	VQIPManager_DeleteRequest	DeleteRequest	Response
Get	VQIPManager_GetRequest	GetRequest	GetResponse
PageSearch	VQIPManager_PageSearchRequest	PageSearchRequest	PageSearchResponse
Share	VQIPManager_AssociateRequest	AssociateRequest	Response
Unshare	VQIPManager_DissociateRequest	DissociateRequest	Response
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

DNS View - zones attributes

DNS_VIEW_ZONE_REC

Following table for **DNS_VIEW_ZONE_REC** lists the required and optional attributes for the actions Add and Delete, and for the actions Get and Update when working with a view specific zone

Table 2-98 DNS_VIEW_ZONE_REC attributes

zone

Type:	**qipws:DNS_ZONE_KEY**			
Operation:	Add	Delete	Get	Update
Optional/ Required:	R	R	R	R
Description:	The key to identify a zone.			

defaultTtl

Type:	**xsd:long**			
Operation:	Add	Delete	Get	Update
Optional/ Required:	O	N/A	N/A	O
Description:	DNS default TTL.			

dnsOptionList

Type:	**qipws:ARRAY_OF_DNS_OPTIONS**			
Operation:	Add	Delete	Get	Update
Optional/ Required:	O	N/A	N/A	O
Description:	List of zero, or more, DNS options.			

email

Type:	**xsd:string**			
Operation:	Add	Delete	Get	Update
Optional/ Required:	O	N/A	N/A	O
Description:	DNS email address.			

expireTime

Type:	**xsd:long**			
Operation:	Add	Delete	Get	Update
Optional/ Required:	O	N/A	N/A	O
Description:	DNS expire time.			

negativeCacheTtl

Type:	**xsd:long**			
Operation:	Add	Delete	Get	Update
Optional/ Required:	O	N/A	N/A	O
Description:	DNS negative cache TTL.			

optionalAttributeList

Type:	**qipws:UdaList**			
Operation:	Add	Delete	Get	Update
Optional/ Required:	O	N/A	N/A	O
Description:	List of zero, or more, user defined attributes.			

postfixZoneExtension

Type:	**xsd:string**			
Operation:	Add	Delete	Get	Update
Optional/ Required:	O	N/A	N/A	O
Description:	DNS postfix extension.			

prefixZoneExtension

Type:	**xsd:string**			
Operation:	Add	Delete	Get	Update
Optional/ Required:	O	N/A	N/A	O
Description:	DNS pretfix extension.			

refreshTime

Type:	**xsd:long**			
Operation:	Add	Delete	Get	Update
Optional/ Required:	O	N/A	N/A	O
Description:	DNS refresh time.			

retryTime

Type:	**xsd:long**			
Operation:	Add	Delete	Get	Update
Optional/ Required:	O	N/A	N/A	O
Description:	DNS retry time.			

DNS_SHARED_ZONE_REC

The following table for **DNS_SHARED_ZONE_REC** applies to the actions Get, Share, Unshare, and Update.

Note: As most of the attributes of the global zone are defined by the global zone itself, there are many values that are read-only.

Table 2-99 DNS_SHARED_ZONE_REC attributes

zone

Type:	**qipws:DNS_ZONE_KEY**			
Operation:	Get	Share	Unshare	Update
Optional/ Required:	R	R	R	R
Description:	The key to identify a zone.			

dnsOptionList

Type:	**qipws:ARRAY_OF_DNS_OPTIONS**		
Operation:	Get	Share	Unshare
Optional/ Required:	O	O	O
Description:	If unpopulated, then the existing options defined in the global zone are inherited. If populated with 0 ore more options, then this list overrides the global zone's options. Refer to Table 2-90, “qipws:ARRAY_OF_DNS_OPTIONS attributes” (p. 149) for more information.		

defaultTtl

Type:	**xsd:long**			
Operation:	Get	Share	Unshare	Update
Optional/ Required:	N/A	N/A	N/A	N/A
Description:	Read-only attributes from global zone.			

email

Type:	**xsd:string**			
Operation:	Get	Share	Unshare	Update
Optional/ Required:	N/A	N/A	N/A	N/A
Description:	Read-only attributes from global zone.			

expireTime

Type:	**xsd:long**			
Operation:	Get	Share	Unshare	Update
Optional/ Required:	N/A	N/A	N/A	N/A
Description:	Read-only attributes from global zone.			

negativeCacheTtl

Type:	**xsd:long**			
Operation:	Get	Share	Unshare	Update
Optional/ Required:	N/A	N/A	N/A	N/A
Description:	Read-only attributes from global zone.			

refreshTime

Type:	**xsd:long**			
Operation:	Get	Share	Unshare	Update
Optional/ Required:	N/A	N/A	N/A	N/A
Description:	Read-only attributes from global zone.			

retryTime

Type:	**`xsd:long`**			
Operation:	Get	Share	Unshare	Update
Optional/ Required:	N/A	N/A	N/A	N/A
Description:	Read-only attributes from global zone.			

optionalAttributeList

Type:	**`qipws:UdaList`**			
Operation:	Get	Share	Unshare	Update
Optional/ Required:	N/A	N/A	N/A	N/A
Description:	Read-only attributes from global zone.			

DNS_ZONE_KEY

The following table lists the attributes for the **`DNS_ZONE_KEY`**.

Table 2-100 DNS_ZONE_KEY attributes

Part name	Type	Opt/Req	Description
`zoneName`	**`xsd:string`**	R	Required for both a zone that is specific to a view, and a global zone that is being shared with this view.
`viewName`	**`xsd:string`**	R	View's name.
`sharedIntoView`	**`xsd:boolean`**	See the description	If false (or unset), it indicates the named zone is specific to the named view. When set to true, it indicates the named zone is a global zone.
`ipv4ReverseZone`	**`xsd:boolean`**	See the description	Must be false (or unset) when sharedIntoView is false. Set to true when the global zone is an IPv4 reverse zone.

DNS View zone - servers

Overview

Purpose

This section documents operations to manage primary and secondary servers for a DNS Zone that is specific to a view.

Table 2-101 DNS View zone - servers operations and messages

Action	Operation	Incoming message	Outgoing message
Add	VQIPManager_AddRequest	AddRequest	Response
Delete	VQIPManager_DeleteRequest	DeleteRequest	Response
Get	VQIPManager_GetRequest	GetRequest:	GetResponse
PageSearch	VQIPManager_PageSearchRequest	PageSearchRequest	PageSearchResponse
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

DNS View zone - servers attributes

DNS_ZONE_SERVER_REC

DNS_ZONE_SERVER_REC holds the attributes of server, which is either a primary or secondary DNS server. For the case of a secondary server, the attribute named **primaryServer** is populated; otherwise, it is left unset.

Table 2-102 DNS_ZONE_SERVER_REC attributes

name

Type:	**xsd:string**				
Operation:	Get	Search	Add	Update	
Optional/ Required:	R	R	R	R	
Description:					

zone

Type:	**qipws:DNS_ZONE_KEY**			
Operation:	Get	Search	Add	Update
Optional/ Required:	R	R	R	R
Description:	Identifies the zone. Refer to Table 2-100, "DNS_ZONE_KEY attributes" (p. 160) for more information.			

dnsOptionList

Type:	**qipws:ARRAY_OF_DNS_OPTIONS**			
Operation:	Get	Search	Add	Update
Optional/ Required:	N/A	N/A	O	O
Description:	Server specific DNS options.			

primaryServer

Type:	**xsd:string**			
Operation:	Get	Search	Add	Update
Optional/ Required:	N/A	N/A	O	O
Description:	Used to associate a secondary server with a primary DNS server.			

SEARCH_DNS_ZONE_SERVER_REC

The following table describes **SEARCH_DNS_ZONE_SERVER_REC** attribute.

Table 2-103 SEARCH_DNS_ZONE_SERVER_REC attribute

Part name	Type	Opt/Req	Description
zone	**qipws:DNS_ZONE_KEY**	**R**	Identifies the zone. Refer to Table 2-100, "DNS_ZONE_KEY attributes" (p. 160) for more information.

DNS_ZONE_SERVER_KEY

The following table describes **DNS_ZONE_SERVER_KEY** attribute.

Table 2-104 DNS_ZONE_SERVER_KEY attribute

Part name	Type	Opt/Req	Description
name	**xsd:string**	R	Name of the server.
zone	**qipws:DNS_ZONE_KEY**	R	Identifies the zone. Refer to Table 2-100, "DNS_ZONE_KEY attributes" (p. 160) for more information.

Add DNS server to zone

Overview

The northbound system invokes **`VQIPManager_AddRequest`** operation with AddRequest message containing a **`DNS_ZONE_SERVER_REC`** structure for adding a server. Refer to "DNS_ZONE_SERVER_REC" (p. 2-162) more information.

Delete DNS server

The northbound system invokes **`VQIPManager_DeleteRequest`** operation with DeleteRequest message containing a **`DNS_ZONE_SERVER_REC`** structure for uniquely identifying the server. Refer to "DNS_ZONE_SERVER_REC" (p. 2-162) more information.

Get DNS server

Overview

The northbound system invokes **`VQIPManager_GetRequest`** operation for retrieving attributes of a specific DNS Server. This operation uses **`GetRequest`** message with **`DNS_ZONE_SERVER_REC`** structure. Refer to "DNS_ZONE_SERVER_REC" (p. 2-162) more information. If the requested server is found, then its attributes are returned. If not found, then an empty record is returned in the **`Response`** message.

PageSearch DNS servers

Overview

The northbound system invokes **`VQIPManager_PageSearchRequest`** operation to get list of servers matching the specified search criteria. This operation uses **`PageSearchRequest`** Message described in "Request structure - MoveObjectRequest message" (p. 2-26). **`SEARCH_DNS_ZONE_SERVER_REC`** structure shall be used for searching address blocks matching the specified search criteria. Refer to "SEARCH_DNS_ZONE_SERVER_REC" (p. 2-163) more information.

Update DNS server

Overview

The northbound system invokes **`VQIPManager_UpdateRequest`** operation with UpdateRequest message containing a **`DNS_ZONE_SERVER_REC`** structure for updating attributes. Refer to "DNS_ZONE_SERVER_REC" (p. 2-162) more information.

For the Update operation, all data fields for a record, those that are being updated as well as those that are not being updated should be sent in as input. Fields for which values are being nulled will be sent in as the attribute name with no values associated with it.

INFO - configuration files

Overview

Purpose

This API provides for reading configuration files from the server.

Table 2-105 INFO - configuration files operation and configuration

Action	Operation	Incoming message	Outgoing message
Get	VQIPManager_GetRequest	GetRequest	Response

INFO - configuration files attributes

INFO_CONFIGFILE_REC

The following table describes the **INFO_CONFIGFILE_REC** attributes.

Table 2-106 INFO_CONFIGFILE_REC attributes

Part name	Type	Get	Description
name	**xsd:string**	R	Identifier for the configuration file. As of VitalQIP 7.2, only *dns_zone_options.xml* is supported.
contents	**xsd:string**	N/A	

Internet Registry

Overview

Purpose

Four Regional Internet Registry (RIR) are supported in VitalQIP: AfriNIC, ARIN, APNIC and RIPE. The following operations are supported for maintaining registry properties:

- Get
- Update

Messages required for each operations are given the following table.

Table 2-107 Internet Registry operations and messages

Action	Operation	Incoming message	Outgoing message
Get	VQIPManager_GetRequest	GetRequest	GetResponse
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

`REGISTRY_REC` structure

The **`REGISTRY_REC`** structure is used for maintaining internet registry properties. The following table lists the required and optional attributes for operations on maintainers.

Table 2-108 REGISTRY_REC attributes

name

Type:	**`qipws:REGISTRY_TYPE`**	
Operation:	Get	Update
Optional/Required:	R	R
Description:	AFRINIC, ARIN, APNIC or RIPE registry.	

emailAddress

Type:	**`xsd:string`**	
Operation:	Get	Update
Optional/Required:	N/A	R
Description:	The e-mail address where reports are sent. The initial default should not be changed except for testing purposes.	

ccList

Type:	**xsd:string**	
Operation:	Get	Update
Optional/Required:	N/A	O
Description:	Additional e-mail addresses where reports are sent.	

minIPv4BlockSize

Type:	**xsd:short**	
Operation:	Get	Update
Optional/Required:	N/A	R
Description:	Minimum IPV4 Block Size used for reporting purposes (defaults to /29).	

minIPv6BlockSize

Type:	**xsd:short**	
Operation:	Get	Update
Optional/Required:	N/A	R
Description:	Minimum IPV6 Block Size used for reporting purposes (defaults to /48).	

whoIsServerName

Type:	**xsd:string**	
Operation:	Get	Update
Optional/Required:	N/A	O
Description:	The default server name for the Internet Registry. This specifies what server to check for “WHOIS” database block information. This field is the FQDN of the Whois Server. The initial default should not be changed except for testing purposes.	

source

Type:	**xsd:string**	
Operation:	Get	Update
Optional/Required:	N/A	O
Description:	Source of data for the Internet Registry.	

Update Registry

Overview

The northbound system invokes **`VQIPManager_UpdateRequest`** operation with UpdateRequest message containing **`REGISTRY_RECDetails`** regarding required and optional attributes are given in "REGISTRY_REC attributes" (p. 2-171).

Note: For the Update operation, all data fields for a record, those that are being updated as well as those that are not being updated, should be sent in as input. Fields for which values are being nulled are sent in as the attribute names with no values associated with them.

Get Registry

Overview

The northbound system invokes **`VQIPManager_GetRequest`** operation for retrieving attributes of a specific internet registry. This operation uses **`GetRequest`** message with **`REGISTRY_REC`** structure. If the specified internet registry is found, attributes of the internet registry are returned. If the internet registry is not found, an empty record is returned in the **`Response`** message.

Maintainer

Overview

Purpose

The maintainer is the person responsible for tracking which addresses in the block are used. At least one maintainer must be defined for each Internet Registry being used. Maintainer is needed when registering a block of addresses. The following operations are supported on maintainer records:

- Add
- Delete
- Get
- Search
- Update

Messages required for each operations are given the following table.

Table 2-109 Maintainer operations and messages

Action	Operation	Incoming message	Outgoing message
Add	VQIPManager_AddRequest	AddRequest	Response
Delete	VQIPManager_DeleteRequest	DeleteRequest	Response
Get	VQIPManager_GetRequest	GetRequest	GetResponse
Search	VQIPManager_SearchRequest	SearchRequest	SearchResponse
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

MAINTAINER_REC structure

The **`MAINTAINER_REC`** structure is used for operations on maintainer records. The following table lists the required and optional attributes for operations on maintainers.

Table 2-110 MAINTAINER_REC attributes

name

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	R	R	N/A	R
Description:	Name or other identifier for the maintainer (up to 10 alphanumeric characters).				

registry

Type:	**qipws:REGISTRY_TYPE**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	N/A	N/A	R	O
Description:	Registry the maintainer is for.				

emailAddress

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	N/A	N/A	N/A	R
Description:	E-mail address of the maintainer (up to 255 characters).				

prefix

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Prefix of the network name in the RIR report (up to 32 alphanumeric characters).				

description

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Text description for the maintainer (up to 256 alphanumeric characters).				

encryptionType

Type:	**qipws:ENCRYPTION_TYPE**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	N/A	N/A	N/A	R
Description:	Required for add/update operations when registry is AfriNIC, APNIC or RIPE registry. N/A for ARIN registry.				

password

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	N/A	N/A	N/A	R
Description:	Required for add/update operations when registry is AfriNIC, APNIC or RIPE registry. N/A for ARIN registry.				

Add maintainer

Overview

The northbound system invokes **`VQIPManager_AddRequest`** operation with AddRequest message containing **`MAINTAINER_REC`** structure for adding a maintainer.

Delete maintainer

Overview

The northbound system invokes **`VQIPManager_DeleteRequest`** operation with DeleteRequest message containing **`MAINTAINER_REC`** structure for deleting a maintainer.

Update maintainer

Overview

The northbound system invokes **`VQIPManager_UpdateRequest`** operation with UpdateRequest message containing **`MAINTAINER_REC`** structure for updating attributes of a maintainer.

Note: For the Update operation, all data fields for a record, those that are being updated as well as those that are not being updated, should be sent in as input. Fields for which values are being nulled are sent in as the attribute name with no values associated with it. Additionally, an Update operation cannot be used for modifying the registry with which the maintainer is associated. To change the registry with which the maintainer is associated, the original maintainer must be deleted first and then added using the new registry type.

Get maintainer

Overview

The northbound system invokes **`VQIPManager_GetRequest`** operation for retrieving a specific maintainer. This operation uses **`GetRequest`** message with **`MAINTAINER_REC`** structure for maintainer. If the specified maintainer is found, the attributes of the maintainer are returned. If the maintainer is not found, an empty record is returned in the **`Response`** message.

Search maintainer

The northbound system invokes **`VQIPManager_SearchRequest`** operation using **`MAINTAINER_REC`** structure for retrieving a list of all maintainers for a registry. This operation uses **`SearchRequest`** message with **`MAINTAINER_REC`** structure containing the registry for which to search maintainers. If maintainers for the specified registry are found, the attributes of the all maintainers in the registry are returned as an array in **`SearchResponse`** message. If no maintainers are defined for the specified registry, an empty array is returned in **`SearchResponse`** message.

Managed files

Overview

Purpose

VitalQIP provides Managed Files as a general mechanism to centrally manage files distributed to remote servers. Managed Files can be used to distribute callouts, scripts, zone files, bind include files, or any other text based files to remote servers. Managed Files can be used in conjunction with server side callouts to replace or augment the VitalQIP generated server configuration.

While VitalQIP does not enforce a maximum file size for a Managed File, performance degrades if the Managed File is larger than 2M. Consider enabling gzip compression on your http(s) session to help reduce the network bandwidth consumed by your SOAP messages.

The following table documents the basic operations available to administer managed files. Because server association data is a component of these messages, it is possible to implement all administrative functionality within this basic set. Nonetheless, a set of redundant actions are provided to simplify managing just server associations.

Table 2-111 Managed files basic operations and messages

Action	Operation	Incoming message	Outgoing message
Add	VQIPManager_AddRequest	AddRequest	Response
Delete	VQIPManager_DeleteRequest VQIPManager_DeleteForcedRequest	DeleteRequest DeleteForcedRequest	Response Response
Get	VQIPManager_GetRequest	GetRequest:	GetResponse
PageSearch	VQIPManager_PageSearchRequest	PageSearchRequest	PageSearchResponse
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

Convenience operations for managing server associations are described in the following table.

Table 2-112 Managed files convenience operations and messages

Action	Operation	Incoming message	Outgoing message
Associate	VQIPManager_AssociateRequest	AssociateRequest	Response
Count	VQIPManager_CountAssociations Request	CountAssociationsRequest	CountAssociationsResponse

Action	Operation	Incoming message	Outgoing message
Dissociate	VQIPManager_DissociateRequest	DissociateRequest	Response
Find	VQIPManager_FindAssociations Request	FindAssociationsRequest:	FindAssociationsResponse

Managed file attributes

MANAGED_FILE_REC

The following table describes the attributes for **MANAGED_FILE_REC**.

Table 2-113 MANAGED_FILE_REC attributes

name

Type:	**xsd:string**			
Operation:	Add	Delete	Get	Update
Optional/Required:	R	R	R	R
Description:	Name or other identifier for the maintainer (up to 10 alphanumeric characters).			

optionalAttributeList

Type:	**qipws:UdaList**			
Operation:	Add	Delete	Get	Update
Optional/Required:	O	N/A	N/A	O
Description:	List of user defined attributes.			

serverList

Type:	**qipws:ARRAY_OF_MANAGED_FILE_SERVERS**			
Operation:	Add	Delete	Get	Update
Optional/Required:	O	N/A	N/A	O
Description:	Server associations.			

binaryData

Type:	**xmime:base64Binary**			
Operation:	Add	Delete	Get	Update
Optional/Required:	O	N/A	N/A	O
Description:	Managed File data.			

ARRAY_OF_MANAGED_FILE_SERVERS

The following table describes the attributes for **ARRAY_OF_MANAGED_FILE_SERVERS**.

Table 2-114 ARRAY_OF_MANAGED_FILE_SERVERS attribute

Part name	Type	Description
serverData	**qipws:MANAGED_FILE_SERVER**	A list.

MANAGED_FILE_SERVER

The following table describes the attribute for **MANAGED_FILE_SERVER**.

Table 2-115 MANAGED_FILE_SERVER attribute

Part name	Type	Description
name	**xsd:string**	Server's name.
type	**qipws:MANAGED_FILE_SERVER_TYPE**	DNS or DHCP.

MANAGED_FILE_KEY

The following table describes the attribute for **MANAGED_FILE_KEY**.

Table 2-116 MANAGED_FILE_KEY attribute

Part name	Type	Opt/Req	Description
name	**xsd:string**	R	Identifies the server.

SEARCH__MANAGED_FILE_REC

The following table describes the attribute for **MANAGED_FILE_KEY**.

Table 2-117 MANAGED_FILE_KEY attributes

Part name	Type	Opt/Req	Description
name	**xsd:string**	O	Name of the managed file.
optionalAttributeList	**qipws:UdaList**	O	List of user defined attributes.
server	**xsd:string**	O	Name of an associated server.

AddRequest message

Overview

The northbound system invokes a **`VQIPManager_AddRequest`** operation with **`AddRequest`** message as described in "Request structure - AddRequest message" (p. 2-10). However, the messages used by this feature can be viewed as being comprised of two parts:

1. The meta-data which contains information about the file such as its name. **`MANAGED_FILE_REC`** is used to hold the meta-data.
2. The file content is attached to the **`MANAGED_FILE_REC`** by the SOAP's MTOM mechanism. Refer to your SOAP toolkit documentation for API details.

Note: A managed file is expected to contain printable characters.

Here is a rough outline of the additional steps to attach file data when using AXIS2:

```
options = mgr._getServiceClient().getOptions();
options.setProperty(
  org.apache.axis2.Constants.Configuration.ENABLE_MTOM,
  org.apache.axis2.Constants.VALUE_TRUE);

// Increase the timeout for large attachments because the server
// needs more processing time.
options.setTimeOutInMilliSeconds(seconds*1000);

FileDataSource fileDataSrc = new FileDataSource(myFile);
ContentType_type0 contentType = new ContentType_type0();
contentType.setContentType_type0(fileDataSrc.getContentType());
DataHandler dataHandler = new DataHandler(fileDataSrc);
Base64Binary base64Binary = new Base64Binary();
base64Binary.setBase64Binary(dataHandler);
base64Binary.setContentType(contentType);
managedFileRec.setBinaryData(base64Binary);
```

AssociateRequest message

Overview

Servers can be associated with a managed file when initially adding the managed file, by an update request, or individually using **`VQIPManager_AssociateRequest`** with an **`AssociateRequest`**. Details of **`AssociateRequest`** message are given in "Request structure - AssociateRequest message" (p. 2-11). **`srcKey`** will be of type **`MANAGED_FILE_SERVER_KEY`** and **`targetKey`** will be of type **`MANAGED_FILE_KEY`**.

CountAssociationsRequest message

Overview

`VQIPManager_CountAssocationsRequest` operation should be used for counting the number of servers associated with a managed file. Details are in "Request structure - CountAssociations message" (p. 2-14). **`objectKey`** will be of type **`MANAGED_FILE_KEY`**.

DeleteRequest and DeleteRequestForced messages

Overview

Either a **`VQIPManager_DeleteRequest`** or **`VQIPManager_DeleteForcedRequest`** can be used to delete a managed file. If the managed file still has server associations, the delete will fail. Either remove the associations, or use **`VQIPManager_DeleteForcedRequest`** with **`forced=true`**.

`VQIPManager_DeleteRequest` takes a **`MANAGED_FILE_REC`**, and **`VQIPManager_DeleteForcedRequest`** takes a **`MANAGED_FILE_KEY`**. Refer to "Request structure - DeleteRequest message" (p. 2-16) and "Request structure - DeleteRequestForced message" (p. 2-17) for more information.

DissociateRequest message

Overview

`VQIPManager_DissociateRequest` operation may be used for dissociating a single server from a managed file. Details of **`DissociateRequest`** message are given in "Request structure - DissociateRequest message" (p. 2-18). **`srcKey`** will be of type **`MANAGED_FILE_SERVER_KEY`** and targetKey will be of type **`MANAGED_FILE_KEY`**. Alternatively, an **`UpdateRequest`** may be used.

FindAssociationsRequest message

Overview

`VQIPManager_FindAssociationsRequest` operation may be used for retrieving the list of servers associated with a managed file. Details of **`FindAssociationRequest`** message are given in "Request structure - FindAssociationsRequest message" (p. 2-20). **`objectKey`** will be of type **`MANAGED_FILE_KEY`**. If the managed file identified by objectKey has any associated rules, objects of type **`MANAGED_FILE_SERVER_KEY`** will be returned.

GetFileRequest Message

Overview

`VQIPManager_GetFileRequest` retrieves both the file's meta data in **`MANAGED_FILE_REC`** and a MIME attachment containing the actual file contents. Refer to your SOAP implementation's MTOM documentation on how to access the MIME attachment. Refer to "Request structure - GetFileRequest message" (p. 2-23) for details regarding a **`GetFileRequest`** message.

Note: The data can be large; possibly too large to fit in memory. Consider using streaming I/O, or other techniques, to reduce memory footprint.

GetRequest message

Overview

Use **`VQIPManager_GetRequest`** with a **`MANAGED_FILE_REC`** to only retrieve information about the managed file, not the file itself. Details are given in "Request structure - GetRequest message" (p. 2-24).

PageSearchRequest message

Overview

Use a **`PageSearchRequest`** with a **`SEARCH_MANAGED_FILE_REC`** object, to search for managed files. **`SEARCH_MANAGED_FILE_REC`** does not have any mandatory arguments, but if searching by a server's name, then the server type must also be specified. Refer to "Request structure - MoveObjectRequest message" (p. 2-26) for more information.

UpdateRequest message

Overview

The northbound system invokes **`VQIPManager_UpdateRequest`** operation with **`UpdateRequest`** containing a **`MANAGED_FILE_REC`** structure for updating an address range. The details are in "Request structure - UpdateRequest message" (p. 2-30).

Nodes

Overview

Purpose

A node represents an IP-enabled device and encompasses a set of interfaces on one or more subnets. The purpose of the node in VitalQIP is to capture the information such as administrative data, grouping, location, and so on. Each interface within a node includes a set of zero or more IP addresses. Each IPAddress includes a set of domin information records. In summary, there are three levels of hierarchy within Nodes.

Node → Interface → IP Address → Domain Information records

The following operations are supported at each level.

Table 2-118 Node operations

Level	Available operations
Node	Add, Delete, Get, Search, Update
Interface	Add, Delete, Move to a different node, Link Object, Unlink Object
IPAddress	Add, Delete, Link, Move, Unlink to a different interface, Move to a different Subnet
DomainInfo	Add, Delete

Messages required for each operation are given in the following table.

Table 2-119 Node operation and messages

Action	Operation	Incoming message	Outgoing message
Add	VQIPManager_AddRequest	AddRequest	Response
AddContained Object	VQIPManager_AddContained Object	AddContainedObject	Response
Delete	VQIPManager_DeleteRequest	DeleteRequest	Response
DeleteContained Object	VQIPManager_DeleteContained Object	DeleteContainedObject Request	Response
Get	VQIPManager_GetRequest	GetRequest	GetResponse
Paginated Search	VQIPManager_PageSearchRequest	PageSearchRequest	PageSearchResponse
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

Add, delete and update node operations

Overview

The northbound system invokes:

- **`VQIPManager_AddRequest`** method for Add
- **`VQIPManager_DeleteRequest`** method for Delete operation
- **`VQIP_GetRequest`** method for Get operation
- **`QIPManager_UpdateRequest`** method for Update operations

Add, Delete and Get operations use **`AddRequest`** message, **`DeleteRequest`** message and **`UpdateRequest`** messages, respectively. For more information, refer to:

- "Request structure - AddRequest message" (p. 2-10)
- "Request structure - DeleteRequest message" (p. 2-16)
- "Request structure - UpdateRequest message" (p. 2-30)

A **`NODE_REC`** object is used for **`reqObject`** attribute of all three messages. Update operation uses **`UpdateRequest`** messages specified in "Request structure - UpdateRequest message" (p. 2-30) with **`NODE_REC`** object being used for **`reqObject`** attribute in **`UpdateRequest`**.

When a node is updated or deleted, the hierarchy of objects associated with the node, such as interface, IP addresses, and so one, is also deleted or updated. For update, the entire hierarchy, including all interfaces, IP addresses, and domains, as they are expected in the updated node must be provided in the input. For delete operation, if a node contains an IPv4 object, the link to the object is removed. **`NODE_REC`** object with attributes that are required/optional at the Node level are given in the following table.

Table 2-120 NODE_REC object attributes

name

Type:	**`xsd:string`**		
Operation:	Add	Delete	Update
Optional/Required:	O	O	O
Description:	Node name(upto 512 characters).		

uniqueID

Type:	**`xsd:string`**		
Operation:	Add	Delete	Update
Optional/Required:	O	R	R

Description:	Unique identifier. If the user does not provide input, this is generated.

description

Type:	**xsd:string**		
Operation:	Add	Delete	Update
Optional/Required:	O	O	O
Description:			

nodeType

Type:	**xsd:string**		
Operation:	Add	Delete	Update
Optional/Required:	R	O	R
Description:	Type of object class, for example, WorkStation, PC, and so on.		

interfaces

Type:	**qipws: INTERFACE_REC**		
Operation:	Add	Delete	Update
Optional/Required:	O	O	O
Description:	List of interfaces associated with the node.		

checkAllNameCollisions

Type:	**xsd:boolean**		
Operation:	Add	Delete	Update
Optional/Required:	O	O	O
Description:	Check if there is a collision with existing alias, object or zone name.		

optionalAttributeList

Type:	**qipws:UdaList**		
Operation:	Add	Delete	Update
Optional/Required:	O	O	O
Description:	List of user defined attributes.		

INTERFACE_REC

A node can be added with or without any interfaces. Addition of interfaces at the time of node creation can be done by populationg a list of **INTERFACE_REC**. Details of required and optional attributes for **INTERFACE_REC** objects are described in the following table.

Table 2-121 INTERFACE_REC object attributes

name

Type:	**xsd:string**	
Operation:	Add	Delete
Optional/Required:	R	R
Description:	Name of interface (up to 64 characters).	

macAddress

Type:	**xsd:string**	
Operation:	Add	Delete
Optional/Required:	O	N/A
Description:	MAC address of the interface.	

ipAddrs

Type:	**qipws: IPADDR_REC**	
Operation:	Add	Delete
Optional/Required:	O	N/A
Description:	List of zero or more IPAddresses associated with the interface. **IPADDR_REC** is described below in further detail.	

optionalAttributeList

Type:	**qipws:UdaList**	
Operation:	Add	Delete
Optional/Required:	O	N/A
Description:	List of zero or more user defined attributes.	

IPADDR_REC

An interface can be added with or without any IP adddresses. Addition of new IPv6 addresses or linking of an existing IPv4 address to an interface can be done by populating a list of **IPADDR_REC**. **IPADDR_REC** can be used to specify either an IPv4 object or an IPv6 object. Details of required and optional attributes for **IPADDR_REC** object are given in the following table.

Table 2-122 IPADDR_REC object attributes

ipAddrStr

Type:	**xsd:string**	
Operation:	Add	Delete
Optional/Required:	O	R
Description:	Either IPv4 or IPv6 address string. • **IPv6 Address:** IPv6Address string is required if allocation algorithm is specified. A new IPv6 Object is created in VitalQIP and added to the interface. • **IPv4 Address:** required if linking an existing IPv4 address to an interface. Unlike IPv6 object, the specified IPv4 object must already exist in VitalQIP. An error is returned if IPv4 object does not already exist.	

ipGroupName

Type:	**xsd:string**	
Operation:	Add	Delete
Optional/Required:	O	N/A
Description:	MAC address of the interface. N/A when **ipAddrStr** is IPv4 address.	

domainInfo

Type:	**qipws: DOMAIN_INFO_REC**	
Operation:	Add	Delete
Optional/Required:	O	N/A
Description:	List of zero or more IP addresses associated with the interface. **DOMAIN_INFO_REC** is described below in further detail. N/A when **ipAddrStr** is IPv4 address.	

subnetStrtAddress

Type:	**`xsd:string`**	
Operation:	Add	Delete
Optional/Required:	O	N/A
Description:	Start address of the subnet where the IPAddress should be created. For example: 1234:5678:: N/A when **`ipAddrStr`** is IPv4 address.	

allocationAlgorithm

Type:	**`qipws:IPADDR_ALLOC_ALG_TYPE`**	
Operation:	Add	Delete
Optional/Required:	O	N/A
Description:	Allocation algorithm to be used for this IP address. N/A when **`ipAddrStr`** is IPv4 address.	

v4ObjectName

Type:	**`xsd:string`**	
Operation:	Add	Delete
Optional/Required:	N/A	N/A
Description:	N/A in current release.	

optionalAttributeList

Type:	**`qipws:UdaList`**	
Operation:	Add	Delete
Optional/Required:	O	N/A
Description:	List of zero or more user defined attributes.	

DOMAIN_INFO_REC

An IPv6 Address can be added to an interface with or without any domain information records. Addition of domain information records can be done by populating a list of **`DOMAIN_INFO_REC`**. Details of required and optional attributes for a **`DOMAIN_INFO_REC`** object are given in the following table.

Table 2-123 DOMAIN_INFO_REC object attributes

fqdn

Type:	**`xsd:string`**	
Operation:	Add	Delete
Optional/Required:	R	R
Description:	Fully Qualified Domain Name of the IPv6 Adress with which this object is associated.	

TTL

Type:	**`xsd:int`**	
Operation:	Add	Delete
Optional/Required:	O	N/A
Description:	Time To Live.	

pubFwdZone

Type:	**`xsd: boolean`**	
Operation:	Add	Delete
Optional/Required:	O	N/A
Description:	Publish domain information in forward zone.	

pubRevZone

Type:	**`xsd: boolean`**	
Operation:	Add	Delete
Optional/Required:	O	N/A
Description:	Publish domain information in reverse zone.	

checkAllNameCollisions

Type:	**`xsd: boolean`**	
Operation:	Add	Delete
Optional/Required:	O	N/A
Description:	Check if there is a collision with existing alias, object or zone name.	

Add an interface, IPv6 address or domain information record

Overview

The northbound system can add Individual interface, IPv6 address and domain information records after a node has been created by using the **`VQIPManager_AddContainedObjectRequest`** message operation using **`AddContainedObject`** described in "Response structures" (p. 2-31). Given below are details related to objects for each type of Add.

Table 2-124 Add objects

object	parentKey	containedObject
interface	NODE_KEYS	INTERFACE_REC
IPv6 address	INTERFACE_KEYS	IPADDR_REC
Domain information record	IPADDR_KEYS	DOMAIN_INFO_REC

Details of each key object is given in Chapter 3.

Note: A separate operation is provided to link/unlink IPv4 object to an interface. Refer to "Link existing IPv4 object to an interface" (p. 2-207) and "Unlink IPv4 object from an interface" (p. 2-208) for details of this operation.

Delete an individual interface, IPv6 address or domain information record

Overview

The northbound system can delete individual interface, IPv6 address or domain information records by invoking **`VQIPManager_DeleteContainedObject`** operation using **`DeleteContainedObject`** message described in "Request structure - DeleteContainedObjectRequest message" (p. 2-15).

The following table provides details on objects required for each deletion:

Table 2-125 Deletion objects

Object	objectkey
Interface	INTERFACE_KEYS
IP address	IPADDR_KEYS
Domain information record	DOMAIN_KEYS

If an interface containing IP objects is deleted, depending on the object type, one of the following two actions are taken on the object:

- If the object is an IPv6 object, the IPv6 object is deleted from VitalQIP.
- If the object is an IPv4 object, only the link to the IPv4 object is removed. The IPv4 object remains in VitalQIP.

Move an interface to another node or an IPv6 Address to another interface

Overview

The northbound system can move an interface to a different node by invoking **`VQIPManager_MoveObjectRequest`** operation. **`MoveObjectRequest`** described in "Request structure - MoveObjectRequest message" (p. 2-26) should be used to move an object.

The following table provides details on objects needed to move an interface and an IP address.

Table 2-126 Move objects

Operation	objKey	toKey
Move an interface to a different node	INTERFACE_KEYS	NODE_KEYS
Move an IPv6 Address to a different interface	IPADDR_KEYS	INTERFACE_KEYS

Link existing IPv4 object to an interface

Overview

The northbound system can manually link (associate) an existing IPv4 object to an interface by invoking **`VQIPManager_LinkAddressToInterfaceRequest`** operation. This operation uses **`LinkAddressToInterfaceRequest`** message. The details of this message are given in the following table.

Table 2-127 LinkAddressToInterfaceRequest message structure

Part Name	Type	Opt / Req	Default	Description
`commonInfo`	**`CommonInfo`**	R	None	Contains user organization and locale. Please refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about CommonInfo structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	N/A in this release.
`ipAddrStr`	**`xsd:string`**	R	None	IPv4 address of the IP object to be linked.
`interfaceKey`	**`INTERFACE_KEYS`**	R	None	Interface to link the IP object to.

This link between IPv4 object and the interface is removed when one of the following three actions occur:

- Node containing an interface with the link to IPv4 object is deleted
- Interface containing the link to IPv4 object is deleted
- IPv4 object is manually unlinked from an interface.

Unlink IPv4 object from an interface

Overview

The northbound system can manually unlink (disassociate) an existing IPv4 object from an interface by invoking **`VQIPManager_UnlinkAddressFromInterfaceRequest`** operation. This operation uses **`UnlinkAddressFromInterfaceRequest`** message. The details of this message are given in the following table.

Table 2-128 UnlinkAddressFromInterfaceRequest message structure

Part name	Type	Opt/ Req	Default	Description
`commonInfo`	**`CommonInfo`**	R	None	Contains user organization and locale. Refer to Table 2-2, "CommonInfo parameters" (p. 2-7) for details about **`CommonInfo`** structure.
`jobReq`	**`qipws:JOB_REQUEST`**	O	None	N/A in this release.
`ipAddrStr`	**`xsd:string`**	R	None	IPv4 address of the IP object being unlinked.
`interfaceKey`	**`qipws:INTERFACE_KEYS`**	R	None	Interface from which to unlink the IP object.

Search and get node

Overview

These operations are supported at the Node level. The northbound system invokes **`VQIPManager_PageSearchRequest`** operation using **`PageRequest`** message and **`SEARCH_NODE_REC`** structure for retrieving a list of one or more node records matching the selection criteria specified in the **`SEARCH_NODE_REC`**. The northbound system invokes **`VitalQIP_GetRequest`** operation using **`Request`** message and **`SEARCH_NODE_REC`** structure to retrieve a single record. Refer to Table 2-15 and Table 2-21 for details on **`GetRequest`** and **`PageSearchRequest`**. Information regarding required and optional attributes of **`SEARCH_NODE_REC`** in **`PageSearchRequest`** are given below. At least one search criteria must be specified. For search, the results are sorted on the value of uniqueID. Wild cards are supported for search operation only.

Search operation supports two sets of mutually exclusive search mode:

- **IPv4 mode:** Finds nodes with IPv4 objects matching the specified IPv4 attributes.
- **IPv6 mode:** Finds nodes with objects (nodes, interfaces, ip objects, domains) matching the specified attributes.

For both the IPv4 and IPv6 search modes, the search primarily returns the matching key values. For example, if searching by an address, such as “192.168.*”, only the matching IP addresses and the associated interface objects are returned. To view all the interfaces and IP addresses associated with a node, it is necessary to perform a separate node get operation. In IPv4 search mode it is possible to search an IPv4 object’s name, MAC Address or UDF value, but those attributes are not included in the output. Since an IPv4 address is just a link, only the link itself is made visible in the node object, not the entire IPv4 object.

Table 2-129 SEARCH_NODE_REC attributes

nodeName

Type:	**`xsd:string`**		
Operation:	Get	IPv6 Search	IPv4 address search
Optional/Required:	O	O	N/A
Description:	Node name.		

nodeType

Type:	**`xsd:string`**		
Operation:	Get	IPv6 Search	IPv4 address search
Optional/Required:	O	O	N/A
Description:	Type of object class for IpV4 for example: WorkStation, PC, and so on.		

uniqueID

Type:	**`xsd:string`**		
Operation:	Get	IPv6 Search	IPv4 address search
Optional/Required:	O	O	N/A
Description:	Unique identifier of node.		

description

Type:	**`xsd:string`**		
Operation:	Get	IPv6 Search	IPv4 address search
Optional/Required:			N/A
Description:	Description.		

nodeUdaList

Type:	**`qipws:UdaList`**		
Operation:	Get	IPv6 Search	IPv4 address search
Optional/Required:	O	O	N/A
Description:	User-defined field at the node level.		

interfaceName

Type:	**`xsd:string`**		
Operation:	Get	IPv6 Search	IPv4 address search
Optional/Required:	O	O	N/A
Description:	Name of one of the interfaces.		

macAddress

Type:	**`xsd:string`**		
Operation:	Get	IPv6 Search	IPv4 address search
Optional/Required:	O	O	O
Description:	IPv6 search mode: MAC Address of one of the interfaces. IPv4 search mode: MAC Address of one of the IPv4 objects linked to the node.		

interfaceUdaList

Type:	**`qipws:UdaList`**		
Operation:	Get	IPv6 Search	IPv4 address search
Optional/Required:	O	O	N/A
Description:	User-defined field at the interface level.		

ipGroupName

Type:	**`xsd:string`**		
Operation:	Get	IPv6 Search	IPv4 address search
Optional/Required:	O	O	N/A
Description:	Name for the IP Address Group (as in Multi-cast address group).		

ipAddress

Type:	**`xsd:string`**		
Operation:	Get	IPv6 Search	IPv4 address search
Optional/Required:	O	O	O
Description:	IPv6 search mode: IPv6 address of one of the IPv6 objects linked to the node. IPv4 search mode: IPv4 address of one of the IPv4 objects linked to the node.		

ipAddressUdaList

Type:	**qipws:UdaList**		
Operation:	Get	IPv6 Search	IPv4 address search
Optional/Required:	O	O	O
Description:	IPv6 search: User-defined attributed at the IP address level. IPv4 search: User-defined field of IPv4 object. Only one user-defined field is allowed.		

fqdn

Type:	**xsd:string**		
Operation:	Get	IPv6 Search	IPv4 address search
Optional/Required:	O	O	N/A
Description:	Fully Qualified Domain Name of an IP address.		

v4ObjectName

Type:	**xsd:string**		
Operation:	Get	IPv6 Search	IPv4 address search
Optional/Required:		N/A	O
Description:	Name of IPv4 object.		

subnetStrtAddr

Type:	**xsd:string**		
Operation:	Get	IPv6 Search	IPv4 address search
Optional/Required:	O	O	N/A
Description:	Start address of a subnet, for example, 1234:5678::		

addressType

Type:	**qipws:V4V6_TYPE**		
Operation:	Get	IPv6 Search	IPv4 address search
Optional/Required:	O	O	O
Description:	Specifies search mode, either IPv4 mode or IPv6 mode. Required for search operation if no other search criteria specified or search type can not be determined from specified criteria.		

Object's view name

Overview

Purpose

The SOAP API documented here covers operations to add and delete an object's fully qualified domain name (FQDN) within a view.

Table 2-130 Object's view name operations and messages

Action	Operation	Incoming message	Outgoing message
fqdnAdd	VQIPManager_AddRequest	AddRequest	Response
fqdnDelete	VQIPManager_DeleteRequest	DeleteRequest	Response
Get	VQIPManager_GetRequest	GetRequest:	GetResponse
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

Object's view name attributes

IPV4_OBJECT_ADDR_KEY

An **UpdateRequest** may use an **IPV4_OBJECT_ADDR_KEY**, though it is unnecessary if the FQDN is not being changed.

Table 2-131 IPV4_OBJECT_ADDR_KEY attribute

Part name	Type	Opt/Req	Description
ipAddrStr	**xsd:string**	R	The IPv4 Address that identifies the object.
fqdn	**xsd:string**	R	A fully qualified domain name.
view	**xsd:string**	R	Name of the view.

IPV4_OBJECT_ADDR_REC

The following table describes attributes for **IPV4_OBJECT_ADDR_REC**.

Table 2-132 IPV4_OBJECT_ADDR_REC attributes

Part name	Type	Opt/Req	Description
ipAddrStr	**xsd:string**	R	The IPv4 Address that identifies the object.
fqdn	**xsd:string**	R	A fully qualified domain name.
view	**xsd:string**	R	Name of the view.
ttl	**xsd:int**	O	DNS Attribute
pubFwdZone	**xsd:boolean**	O	DNS Attribute
checkCollisions	**xsd:boolean**	O	Check if FQDN is unique.

Add an FQDN

Overview

The northbound system invokes **`VQIPManager_AddRequest`** operation with AddRequest message containing an **`IPV4_OBJECT_ADDR_REC`** structure for adding an FQDN to a legacy IPv4 object.

Delete an FQDN

Overview

The northbound system invokes **`VQIPManager_DeleteRequest`** operation with DeleteRequest message containing an I**`PV4_OBJECT_ADDR_REC`** structure for deleting an FQDN name from an IPv4 object.

Get an FQDN

Overview

The northbound system invokes **`VQIPManager_GetRequest`** operation for retrieving FQDN attributes for a specific IPv4 object. This operation uses **`GetRequest`** message with the **`IPV4_OBJECT_ADDR_REC`** structure. If the requested server is found, then its attributes are returned. If not found, then an empty record is returned in the Response message.

Update an FQDN

Overview

The northbound system invokes **`VQIPManager_UpdateRequest`** operation with UpdateRequest message containing an **`IPV4_OBJECT_ADDR_REC`** structure for updating attributes.

Note: For Update operation, all data fields for a record, those that are being updated as well as those that are not being updated should be sent in as input. Fields for which values are being nulled will be sent in as the attribute name with no values associated with it.

Reverse zone templates

Overview

Purpose

Reverse zone templates are used to automatically create a reverse zone when a network is created. Operations supported for reverse zone templates are:

- Add
- Copy
- CountAssociations
- Delete
- Get
- Search
- Update

Messages required for each operation are given in the following table.

Table 2-133 Reverse zone operation and messages

Action	Operation	Incoming message	Outgoing message
Add	VQIPManager_AddRequest	AddRequest	Response
Copy	VQIPManager_CopyRequest	CopyRequest	Response
CountAssociations	VQIPManager_CountAssociations	CountAssociations Request	CountAssociations Response
Delete	VQIPManager_DeleteRequest	DeleteRequest	Response
Get	VQIPManager_GetRequest	GetRequest	GetResponse
Search	VQIPManager_SearchRequest	SearchRequest	SearchResponse
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

Add, delete, get, search and update operations

Overview

VQIPManager_AddRequest operation should be used for adding a subnet profile template. Similarly, **VQIPManager_DeleteRequest** operation should be used for deleting a subnet profile template. **VQIPManager_DeleteRequest** operation should be used for deleting a single reverse zone template. The **VQIPManager_GetRequest** operation should be used to retrieve a single reverse zone template whereas **VQIPManager_SearchRequest** operation should be used to retrieve a list of reverse zone templates. Updates can be done by invoking the **VQIPManager_UpdateRequest** operation, which uses **UpdateRquest** message described in "Request structure - UpdateRequest message" (p. 2-30). **REV_ZONE_TEMPL_REC** should be used for all operations. The attributes of **REV_ZONE_TEMPL_REC** that are required for each operation are described in the following table.

Table 2-134 REV_ZONE_TEMPL_REC attributes

name

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	R	R	O	R
Description:	Name of reverse zone template.				

description

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Further information.				

serverPairs

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Each reverse zone template can have zero or more servers.				

The following tables describe the server pair structure.

Table 2-135 ARRAY_OF_SERVER_PAIR_REC structure

Part name	Type	Opt/ Req	Default	Description
serverPairs	**qipws:SERVER_ PAIR_REC**	O	None	An array of one or more **SERVER_PAIR_REC** objects.

Table 2-136 SERVER_PAIR_REC structure

primaryDnsServer

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Either the primary DNS server or secondary DNS server must be specified.				

secondaryDnsServer

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Either the primary DNS server or secondary DNS server must be specified.				

CopyRequest operation

Overview

The **VQIPManager_CopyRequest** operation should be used to copy an address template. This operation uses **CopyRequest** message, as detailed in the following table.

Table 2-137 CopyRequest message structure

Part name	Type	Opt/ Req	Default	Description
commonInfo	**qipws:CommonInfo**	R	None	Contains user organization and locale.
jobReq	**qipws:JOB_REQUEST**	O	None	N/A in this release.
from	**qipws:VQIP_KEY_ ENTITY**	R	None	Key (name) of reverse zone template from which the data should be copied.
to	**xsd:string**	R	None	Name to be used for the new address template.

CountAssociations message

Overview

The **`VQIPManager_CountAssocationsRequest`** operation should be used to count the number of pools associated with a reverse zone template. **`objectKey`** is of type **`REV_ZONE_TEMPL_KEYS`**.

Rules

Overview

Purpose

Rules allow VitalQIP users to automate the process of allocating blocks per specifications defined in the rule. The following operations are supported for rules:

- Add
- Copy
- Delete
- Get
- Search
- Update

Messages required for each operation are given in the following table.

Table 2-138 Rules operation and messages

Action	Operation	Incoming message	Outgoing message
Add	VQIPManager_AddRequest	AddRequest	Response
Copy	VQIPManager_CopyRequest	CopyRequest	Response
Delete	VQIPManager_DeleteRequest	DeleteRequest	Response
Get	VQIPManager_GetRequest	GetRequest	GetResponse
Search	VQIPManager_SearchRequest	SearchRequest	SearchResponse
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

Add, delete, get, search and update messages

Overview

The northbound system invokes **VQIPManager_AddRequest** operation for Add and **VQIPManager_DeleteRequest** for Delete operation. It invokes **VQIPManager_GetRequest** for the Get operation, **VQIPManager_SearchRequest** for the Search operation and **VQIPManager_UpdateRequest** for the Update functionality using **RULE_REC** structure described in the following table.

Table 2-139 RULE_REC structure attributes

name

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	R	R	O	R
Description:	Unique name of up to 64 alphanumeric characters.				

ruleLevel

Type:	**qipws:RULE_LEVEL**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	N/A	N/A	N/A	O
Description:	Limits what rule is available to administrator for block allocation. Possible values are USED, FREE, SITE and RESERVED.				

ruleType

Type:	**qipws:RULE_TYPE**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	N/A	O	O	O
Description:	Determines type to be used during block allocation. Possible values are USED, FREE, RESERVED and SITE.				

v4V6Rule

Type:	**qipws:V4V6_TYPE**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	N/A	N/A	O	O
Description:	Rule applies to V4 or V6 address space. Possible values are IPv4 and IPv6.				

requestedSize

Type:	**short**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	N/A	N/A	N/A	O
Description:	Determine the size of the block requested by this rule. Valid values for IPV4 are 8-32 and IPV6 are 11-128.				

ruleComponentList

Type:	**qipws:ARRAY_OF_RULE_COMPONENT**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R for **Site** Rule Type	N/A	N/A	N/A	R for **Site** Rule Type
Description:	Required if rule type is site. Not used for any other rule type other than site.				

addrTemplate

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	The name of the object template associated with the rule. Only contiguous used rules can have object templates associated with them.				

subnetProfTemplate

Type:	**`xsd:string`**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	The name of the subnet profile template associated with the rule. Only contiguous used rules can have subnet profile templates associated with them. N/A for IPv6 Rules.				

description

Type:	**`xsd:string`**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Additional information about the rule. Up to 255 alphanumeric characters.				

Get operation

The Get operation retrieves a single record, whereas the search operation should be used to retrieve a list of one or more records. For the Search operation, either **`name`**, **`ruleType`** or **`v4V6Rule`** must be specified.

Search operation

Wildcard character for the Search operation is '*'. For example, when searching for rules that have names that start with "V4Rule", the value of the name attribute could be "V4R*".

Update operation

Please note that for the Update operation, all data fields for a record, including those that are being updated as well as those that are not being updated, should be sent in as input. Fields for which values are being nulled are sent in as the attribute name with no values associated with it. Additionally, an Update operation cannot be used for deletion of individual elements of **`ruleComponentList`**, which is an array of **`RULE_COMPONENT`** objects that is required for rules of type "Site". Please refer to Table 2-140 for details.

Array_Of_Rule_Component

Rules of type Site have one or more associated objects of type **`RULE_COMPONENT`**, as described in the following table.

Table 2-140 RULE_COMPONENT attributes

Part name	Type	Opt/Req	Default	Description
ruleComponentData	**qipws:RULE_ COMPONENT**	R if **RULE_TYPE** is Site. Optional for all other types.	None	An array of one or more **RULE_COMPONENT** objects.

Rule_Component

Overview

A rule component is used to allow specification of subnet block specific rules when the rule type is site. The structure of **RULE_COMPONENT** is given in the following table.

Table 2-141 RULE_COMPONENT structure

Part name	Type	Opt/ Req	Default	Description
addrTemplate	**xsd:string**	O	N/A	The template to be used for creating IP objects in the subnet to be created. Only applicable when a USED block is assigned.
subnetProfTemplate	**xsd:string**	O	N/A	The template for the creating the subnet profile of the subnet to be created. Only applicable when a USED block is assigned. N/A for IPv6 Rules.
ruleType	**qipws:RULE_ COMPONENT_ TYPE**	O	N/A	Status of the block to be created. This is an enumeration type.
subnetBlockSize	**xsd:short**	O	N/A	The CIDR size of the sub-block to be created (that is, /20).

CopyRuleRequest message

Overview

The **VQIPManager_CopyRuleRequest** operation should be used to copy a rule. This operation uses **CopyRuleRequest** message. The details of this message are given in the following table.

Table 2-142 CopyRuleRequest message attributes

Part name	Type	Opt/ Req	Default	Description
commonInfo	**qipws:CommonInfo**	R	None	Contains user organization and locale. Please refer to Table 2-2, "CommonInfo parameters" (p. 7) for details about **CommonInfo** structure.
jobReq	**qipws:JOB_REQUEST**	O	None	N/A in this release.
fromRule	**qipws:VQIP_BASE_ ENTIRY**	R	None	**RULE_REC** from which the data should be copied.
toRule	**xsd:string**	R	None	Name to be used for the new rule.

Subnet profile templates

Overview

Purpose

`VQIPManager_AddRequest` operation should be used for adding a subnet profile template. Similarly, **`VQIPManager_DeleteRequest`** operation should be used for deleting a subnet profile template.

The following operations are available.

- Associate
- Add
- CountAssociations
- Copy
- Dissociate
- Delete
- FindAssociations
- Get
- Search
- Update

The operation (as defined in the WSDL) that needs to be executed for each of the above operations is described in the following table.

Table 2-143 Subnet profile template operation and messages

Action	Operation	Incoming message	Outgoing message
Associate	VQIPManager_AssociateRequest	AssociateRequest	Response
Add	VQIPManager_AddRequest	AddRequest	Response
Copy	VQIPManager_CopyRequest	CopyRequest	Response
CountAssociations	VQIPManager_CountAssociations	CountAssociations Request	CountAssociations Response
Delete	VQIPManager_DeleteRequest	DeleteRequest	Response
Dissociate	VQIPManager_DissociateRequest	DissociateRequest	Response
FindAssociations	VQIPManager_FindAssociations	FindAssociations Request	FindAssociations Response
Get	VQIPManager_GetRequest	GetRequest	GetResponse

Action	Operation	Incoming message	Outgoing message
Search	VQIPManager_SearchRequest	SearchRequest	SearchResponse
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

Add, delete, get, search and update operations

Overview

The northbound system can add and delete subnet profile templates using **`VQIPManager_GenericRequest`** operation using **`Request`** message. The **`VQIPManager_GetRequest`** operation should be used to retrieve a single subnet profile template, whereas **`VQIPManager_SearchRequest`** operation should be used to retrieve a list of subnet profile templates. Updates can be done by invoking the **`VQIPManager_UpdateRequest`** operation, which uses **`UpdateRquest`** message described in "Request structure - UpdateRequest message" (p. 2-30). SUBNET_PROF_TEMPL_REC should be used for all operations. The attributes of SUBNET_PROF_TEMPL_REC that are required for each operation are described in the following table.

Table 2-144 SUBNET_PROF_TEMPL_REC

name

Type:	**`xsd:string`**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	R	R	O	R
Description:	Unique name of up to 64 alphanumeric characters.				

description

Type:	**`xsd:string`**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Additional information.				

domain

Type:	**`xsd:string`**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	N/A	N/A	N/A	R
Description:	Valid values should be determined by using the **`VQIP_Manager_GetAll`** operation.				

dhcpServer

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	N/A	N/A	N/A	R
Description:	Valid values should be determined by using the **VQIP_Manager_GetAll** operation.				

dhcpOptTempl

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	DHCP Option Template to be associated with this scope of addresses. Valid values should be determined by using the **VQIP_Manager_GetAll** operation.				

prefDnsServerI

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	R	N/A	N/A	N/A	R
Description:	Valid values should be determined by using the **VQIP_Manager_GetAll** operation.				

altDnsServerI

Type:	**xsd:string**				
Operation:	Add	Delete	Get	Search	Update
Optional/Required:	O	N/A	N/A	N/A	O
Description:	Valid values should be determined by using the **VQIP_Manager_GetAll** operation.				

AssociateRequest message

Overview

The **VQIPManager_AssociateRequest** operation should be used to associate a subnet profile template with a rule. This operation uses **AssociateRequest** message. Details of **AssociateRequest** message are given in Table 2-6, "AssociateRequest details" (p. 11). **srcKey** is of type **RULE_KEYS** and **targetKey** is of type **SUBNET_PROF_TEMPL_KEYS**.

CopyRequest message

Overview

The **VQIPManager_CopyRequest** operation should be used to copy a subnet profile template. This operation uses **CopyRequest** message. The details of **CopyRequest** message are given in Table 2-137, “CopyRequest message structure” (p. 2-222). The **from** parameter is **SUBNET_PROF_TEMPL_KEYS** and the **to** parameter is the name of the target template.

CountAssociations message

Overview

The **`VQIPManager_CountAssocationsRequest`** operation should be used to count the number of rules associated with a subnet profile template. **`objectKey`** is of type **`SUBNET_PROF_TEMPL_KEYS`**.

DissociateRequest message

The **`VQIPManager_DissociateRequest`** operation should be used to dissociate a subnet profile template from a rule. Details of **`DissociateRequest`** message are given in Table 2-12, “DissociateRequest details” (p. 2-18). **`srcKey`** is of type **`RULE_KEYS`** and **`targetKey`** is of type **`SUBNET_PROF_TEMPL_KEYS`**.

FindAssociationsRequest message

Overview

The **VQIPManager_FindAssociationsRequest** operation should be used to retrieve the list of rules associated with a subnet profile template. Details of **FindAssociationsRequest** message are given in "FindAssociations details" (p. 2-20). **objectKey** is of type **SUBNET_PROF_TEMPL_KEYS**. If the subnet profile template identified by **objectKey** has any associated rules, objects of type **RULE_KEYS** are returned.

UDA definition

Overview

Purpose

Before User Defined Attributes (UDA) can be used, the UDA needs to be defined and then assigned to infrastructure types. A UDA's definition determines, for example, whether the UDA's value contains text or an IP address, whether it's mandatory or optional, and so on. By assigning the UDA to an infrastructure type, it can be used when instances of that infrastructure are created. The following operations are supported.

UDA definition management

Operations that manage a UDA's definition are supported, as described in the following table.

Table 2-145 Manage UDA definitions

Action	Operation	Incoming message	Outgoing message	Description
Add	VQIPManager_AddRequest	AddRequest	Response	Add a new UDA definition.
Delete	VQIPManager_DeleteRequest	DeleteRequest	Response	Delete a UDA's definition.
Get	VQIPManager_GetRequest	GetRequest	GetResponse	Get an existing UDA definition.
Search	VQIPManager_SearchRequest	SearchRequest	SearchResponse	Search for UDA definitions.
Update	VQIPManager_UpdateRequest	UpdateRequest	Response	Update a UDA's definition.

UDA infrastructure association

Operations that associate UDA definitions with infrastructure are supported, as described in the following table.

Table 2-146 Associate UDA with infrastructure

Action	Operation	Incoming message	Outgoing message	Description
Associate	VQIPManager_Associate ListRequest	AssociateList Request	Response	Associate a UDA definition with an infrastructure type.
Dissociate	VQIPManager_Dissociate ListRequest	DissociateList Request	Response	Dissociate a UDA definition from an infrastructure type.

Action	Operation	Incoming message	Outgoing message	Description
Count	VQIPManager_Count AssociationsRequest	CountAssociations Request	Count Associations Response	Count the number of associations with a given infrastructure type.
Find	VQIPManager_Find AssociationsRequest	FindAssociations Request	Find Associations Response	Find the associations with a given infrastructure type.

UDA/UDA group sequencing

Sequencing is supported, as described in the following table.

Table 2-147 Sequence UDAs

Action	Operation	Incoming message	Outgoing message	Description
List	VQIPManager_-GetAssociatedComponents Request	GetAssociated-Components Request	GetAssociated-Components Response	List UDA and UDA group definitions in their assigned sequence.
Sequence	VQIPManager_-SequenceAssociated ComponentsRequest	Sequence Associated-Components Request	Response	Sequence a set of UDA and UDA group definitions to assign the order in which they appear.

Set UDA instance values

Setting the value of a UDA instance is supported, as described in the following table.

Table 2-148 Set UDA values

Action	Operation	Incoming message	Outgoing message	Description
Set	VQIPManager_Set UdaRequest	SetUda Request	Response	Provides a convenient means to set individual UDA values for a wide variety of VitalQIP infrastructure types, without changing a UDA's definition. This capability overlaps with the "update" action provided by most other entities. What distinguishes this operation from "update" is its support for more VitalQIP infrastructure types.

UDA_DEFINITION_REC attributes

The following table lists the required and optional attributes for operations on UDA definition records.

Table 2-149 UDA_DEFINITION_REC attributes

name

Type:	**`xsd:string`**			
Operation:	Add	Delete	Get	Update
Optional/Required:	R	R	R	R
Description:	UDA's name.			

type

Type:	**`qipws:UDA_DEFINITION_ATTRIBUTE_TYPE`**			
Operation:	Add	Delete	Get	Update
Optional/Required:	R	N/A	N/A	R
Description:	The type of this UDA (such as NUMERIC).			

dropDownList

Type:	**`qipws:StringList`**			
Operation:	Add	Delete	Get	Update
Optional/Required:	R	N/A	N/A	R
Description:	An enumeration of the allowed values in a drop-down list. **Note:** **`dropDownList`** is necessary if the value of **`type`** is "LIST".			

initialValue

Type:	**`xsd:string`**			
Operation:	Add	Delete	Get	Update
Optional/Required:	R	N/A	N/A	R
Description:	Initial UDA value. **Note:** Not valid when **`type`** is "LIST".			

readOnly

Type:	**`xsd:boolean`**			
Operation:	Add	Delete	Get	Update
Optional/Required:	R	N/A	N/A	R
Description:	Set to true when UDA is read-only. **Note:** Not valid when **`type`** is "LIST".			

minimum

Type:	**`xsd:int`**			
Operation:	Add	Delete	Get	Update
Optional/Required:	O	N/A	N/A	O
Description:	Only for types "NUMERIC" and "TEXT". If TEXT, then minimum string length. If NUMERIC, then minimum integer.			

maximum

Type:	**`xsd:int`**			
Operation:	Add	Delete	Get	Update
Optional/Required:	O	N/A	N/A	O
Description:	Only for types "NUMERIC" and "TEXT". If TEXT, then minimum string length. If NUMERIC, then minimum integer.			

required

Type:	**`xsd:boolean`**			
Operation:	Add	Delete	Get	Update
Optional/Required:	R	N/A	N/A	R
Description:	Set to true if a UDA instance will require a value. **Note:** Part name **`required`** is necessary if the value of **`readOnly`** is false.			

inheritValue

Type:	**`xsd:boolean`**			
Operation:	Add	Delete	Get	Update
Optional/Required:	R	N/A	N/A	R
Description:	Inherits initial value of the Attribute from its parent item. For example. an Attribute in a block can inherit its initial value from the same Attribute in the block's parent Pool.			

preEditCallout

Type:	**`xsd:string`**			
Operation:	Add	Delete	Get	Update
Optional/Required:	O	N/A	N/A	O
Description:	Name of a callout function which is executed prior to a user editing the Attribute value. **Note:** If a Pre-Edit Callout is selected that will populate an Attribute's initial value and Value Inheritance is activated, then the value associated with the Value Inheritance feature overwrites the value populated by the Pre-Edit Callout.			

postEditCallout

Type:	**`xsd:string`**			
Operation:	Add	Delete	Get	Update
Optional/Required:	O	N/A	N/A	O
Description:	Name of a callout function that is executed prior to saving the Attribute value in the database. If both **`preEditCallout`** and **`postEditCallout`** are selected, the value returned from **`postEditCallout`** is saved to the database.			

validationCallout

Type:	**`xsd:string`**			
Operation:	Add	Delete	Get	Update
Optional/Required:	O	N/A	N/A	O
Description:	Name of a validation callout function. The validation callout function is executed prior to saving the Attribute value. An error is returned if the function returns false. If there is an error, the user will need to modify the Attribute value to comply with the validation.			

UDA_DEFINITION_KEY attribute

The following table describes the **UDA_DEFINITION_KEY** attribute.

Part name	Type	Opt/Req	Description
name	**xsd:string**	R	UDA's name.

UDA_DEFINITION_LIST_REC attribute

The following table describes the UDA_DEFINITION_LIST_REC attribute.

Part name	Type	Opt/Req	Description
uda or **udaGroup**	**xsd:string**	R	Zero, or more, instances of a "uda" or "udaGroup" elements. These elements may be used in combination.

UDA_INFRASTRUCTURE_TYPE_KEY attribute

The following table describes the UDA_INFRASTRUCTURE_TYPE_KEY attribute.

Part name	Type	Opt/Req	Description
name	**qipws:UDA_INFRASTRUCTURE_TYPE**	R	One of the enumerated infrastructure type values.

UDA Set keys

The following table describes the UDA Set keys.

Table 2-150 UDA Set keys

Infrastructure Type	Key	Description
AC_DEVICE	AC_DEVICE_KEY	Refer to "AC_DEVICE_KEY attributes" (p. 2-39).
AC_SUBSCRIBER	AC_SUBSCRIBER_KEY	Refer to "AC_SUBSCRIBER_KEY attributes" (p. 2-43).
BLOCK	ADDRESS_BLOCK_KEYS	Refer to "ADDRESS_BLOCK_KEYS" (p. 3-5)
DC_SERVER DHCP_SERVER DNS_SERVER	SERVER_KEY	Refer to "SERVER_KEY" (p. 3-33).

Infrastructure Type	Key	Description
DNS_VIEW	DNS_VIEW_KEY	Refer to "DNS_VIEW_KEY" (p. 2-153).
INTERFACE	INTERFACE_KEYS	Refer to "INTERFACE_KEYS" (p. 3-13).
IPV4_ADDRESS	IPV4_OBJECT_ADDR_KEY	Refer to "IPV4_OBJECT_ADDR_KEY" (p. 2-214).
IPV4_SUBNET	V4_SUBNET_KEYS	Refer to "V4_SUBNET_KEYS attribute" (p. 2-247).
IPV6_ADDRESS	IPADDR_KEYS	Refer to "IPADDR_KEYS" (p. 3-14).
IPV6_SUBNET	V6_SUBNET_KEYS	Refer to section "V6_SUBNET_KEYS" (p. 3-42).
MANAGED_FILE	MANAGED_FILE_KEY	Refer to "MANAGED_FILE_KEY" (p. 2-186).
MYVIEW	MYVIEW_KEY	Refer to "MYVIEW_KEY attribute" (p. 2-247).
NETWORK	NETWORK_KEY	Refer to "NETWORK_KEY attribute" (p. 2-247)
NODE	NODE_KEYS	Refer to "NODE_KEYS" (p. 3-20).
ORGANIZATION	ORG_KEY	Refer to "ORG_KEY attribute" (p. 2-247).
POOL	POOL_KEYS	Refer to "POOL_KEYS" (p. 3-22).
SUBNET_ ORGANIZATION	SUBNET_ORG_KEY	Refer to "SUBNET_ORG_KEY attribute" (p. 2-247).
ZONE	DNS_ZONE_KEY	Refer to "DNS_ZONE_KEY" (p. 2-160).

MYVIEW_KEY attribute

This key only supports the UDA “set value” operation. The following table describes the MYVIEW_KEY attribute.

Part Name	Type	Opt/Req	Description
name	**xsd:string**	R	Name of a “My View” instance.

NETWORK_KEY attribute

This key only supports the UDA “set value” operation. The following table describes the NETWORK_KEY attribute.

Part Name	Type	Opt/Req	Description
address	**xsd:string**	R	IP Address

ORG_KEY attribute

This key only supports the UDA “set value” operation. The following table describes the ORG_KEY attribute.

Part Name	Type	Opt/Req	Description
name	**xsd:string**	R	Organization name

SUBNET_ORG_KEY attribute

This key only supports the UDA “set value” operation. The following table describes the SUBNET_ORG_KEY attribute.

Part Name	Type	Opt/Req	Description
name	**xsd:string**	R	Name of subnet organization.

V4_SUBNET_KEYS attribute

This key only supports the UDA “set value” operation. The following table describes the V4_SUBNET_KEYS attribute.

Part Name	Type	Opt/Req	Description
address	**xsd:string**	R	IPv4 Address.

UDA group definition

Overview

Purpose

Provides a set of operations for managing UDA groups, which includes their creation and assignment of UDAs to the group. UDAs contained within a group also have an ordering that is represented by their positional sequence within that list.

The following operations are supported.

UDA group definition management

Operations that manage a UDA group's definition are supported, as described in the following table.

Table 2-151 Manage UDA group definitions

Action	Operation	Incoming message	Outgoing message	Description
Add	VQIPManager_AddRequest	AddRequest	Response	Add a new UDA definition.
Delete	VQIPManager_DeleteRequest	DeleteRequest	Response	Delete a UDA's definition.
Get	VQIPManager_GetRequest	GetRequest	GetResponse	Get an existing UDA definition.
Search	VQIPManager_SearchRequest	SearchRequest	SearchResponse	Search for UDA definitions.
Update	VQIPManager_UpdateRequest	UpdateRequest	Response	Update a UDA's definition.

UDA group infrastructure association

Operations that associate UDA group definitions with infrastructure are supported, as described in the following table.

Table 2-152 Associate UDA group definition with infrastructure

Action	Operation	Incoming message	Outgoing message	Description
Associate	VQIPManager_Associate ListRequest	AssociateList Request	Response	Associate a UDA definition with an infrastructure type.
Dissociate	VQIPManager_Dissociate ListRequest	DissociateList Request	Response	Dissociate a UDA definition from an infrastructure type.

Action	Operation	Incoming message	Outgoing message	Description
Count	VQIPManager_Count AssociationsRequest	CountAssociations Request	Count Associations Response	Count the number of associations with a given infrastructure type.
Find	VQIPManager_Find AssociationsRequest	FindAssociations Request	Find Associations Response	Find the associations with a given infrastructure type.

UDA/UDA group sequencing

Sequencing is supported, as described in the following table.

Table 2-153 Sequence UDA/UDA groups

Action	Operation	Incoming message	Outgoing message	Description
List	VQIPManager_-GetAssociatedComponents Request	GetAssociated-Components Request	GetAssociated-Components Response	List UDA and UDA group definitions in their assigned sequence.
Sequence	VQIPManager_-SequenceAssociated ComponentsRequest	Sequence Associated-Components Request	Response	Sequence a set of UDA and UDA group definitions to assign the order in which they appear.

Note: GetAssociatedComponentsResponse contains a list of UDA_DEFINITION_REC and UDAGROUP_DEFINITION_REC in their current sequence.

SequenceAssociatedComponentsRequest shall contain UDA_DEFINITION_REC and UDAGROUP_DEFINITION_REC in the desired sequence.

UDAGROUP_DEFINITION_REC attributes

The following table lists the required and optional attributes for operations on UDA group definition records.

Table 2-154 UDAGROUP_DEFINITION_REC attributes

name

Type:	**`xsd:string`**			
Operation:	Add	Delete	Get	Update
Optional/Required:	R	R	R	R
Description:	UDA's name.			

udaList

Type:	**`qipws:StringList`**			
Operation:	Add	Delete	Get	Update
Optional/Required:	R	N/A	N/A	R
Description:	List of UDA names.			

UDAGROUP_DEFINITION_KEY attributes

The following table lists the required and optional attributes for operations on UDA group definition keys.

Part Name	Type	Opt/Req	Description
`name`	**`xsd:string`**	R	UDA's name.

V6 subnets

Overview

Purpose

A subnet represents the lowest level entity in the network to which an IPv6 address prefix is assigned. The following operations are allowe on an IPv6 subnet:

- Add
- Copy
- Delete
- Get
- Merge
- Renumber
- Search
- Split
- Update

The operation that needs to be executed for each of the above operations is described in the following table.

Table 2-155 V6 subnet operations and messages

Action	Operation	Incoming message	Outgoing message
Add	VQIPManager_AddRequest	AddRequest	Response
Copy	VQIPManager_CopyRequest	CopyRequest	Response
Delete	VQIPManager_DeleteRequest	DeleteRequest	Response
Get	VQIPManager_GetRequest	GetRequest	GetResponse
Merge	VQIPManager_MergeRequest	MergeRequest	Response
Renumber	VQIPManager_RenumberRequest	RenumberRequest	Response
Split	VQIPManager_SplitRequest	SplitRequest	Response
Paginated Search	VQIPManager_PageSearchRequest	PageSearchRequest	PageSearchResponse
Update	VQIPManager_UpdateRequest	UpdateRequest	Response

V6_SUBNET_REC structure

The **V6_SUBNET_REC** structure is used for operations on IPv6 subnets. Following table lists the required and optional attributes of **V6_SUBNET_REC** structure.

Table 2-156 V6_SUBNET_REC attributes

name

Type:	**`xsd:string`**		
Operation:	Add	Delete	Get
Optional/Required:	N/A	N/A	N/A
Description:	Name cannot be specified during Add operation.		

startAddress

Type:	**`xsd:string`**		
Operation:	Add	Delete	Get
Optional/Required:	R	R	R
Description:	Starting address of the subnet in IPV6 format, for example, 1234:5678::.		

description

Type:	**`xsd:string`**		
Operation:	Add	Delete	Get
Optional/Required:	N/A	N/A	N/A
Description:	Additional information. Description cannot be specified during Add operation.		

subnetLength

Type:	**`xsd:short`**		
Operation:	Add	Delete	Get
Optional/Required:	R	N/A	N/A
Description:	Length of the new subnet prefixes to be created, Valid values are 9-128.		

defaultDomain

Type:	**`xsd:string`**		
Operation:	Add	Delete	Get
Optional/Required:	O	N/A	N/A
Description:	Default domain name for all the subnets.		

numTocreate

Type:	**`xsd:short`**		
Operation:	Add	Delete	Get
Optional/Required:	R	N/A	N/A
Description:	Number of subnets to be created.		

allocationType

Type:	**`qipws: SUBNET_ALLOCATION_TYPE`**		
Operation:	Add	Delete	Get
Optional/Required:	R	N/A	N/A
Description:	Determines how the address blocks are allocated within the subnet.		

startType

Type:	**`qipws:START_TYPE`**		
Operation:	Add	Delete	Get
Optional/Required:	R if **`allocationType`** is "EXPLICIT"	N/A	N/A
Description:	Used only when **`allocationType`** is EXPLICIT to determine where to start allocating blocks.		

allocationAlgorithm

Type:	**`qipws: IPADDR_ALLOC_ALG_TYPE`**		
Operation:	Add	Delete	Get
Optional/Required:	R	N/A	N/A
Description:	Determines how the IP address is allocated within the subnet.		

addressTemplateName

Type:	**`xsd: string`**		
Operation:	Add	Delete	Get
Optional/Required:	O	N/A	N/A
Description:	Only IPv6 address template is relevant.		

optionalAttributeList

Type:	**`qipws:UdaList`**		
Operation:	Add	Delete	Get
Optional/Required:	O	N/A	O
Description:	Optional list of User Defined Attributes.		

Add, delete, get and update operations

Overview

The northbound system invokes the following operations to retrieve attributes of IPv6 subnet:

- **`VQIPManager_AddRequest`** operation to add IPv6 subnet
- **`VQIPManager_DeleteRequest`** to delete IPv6 subnet
- **`VQIPManager_GetRequest`** operation to retrieve a single IPv6 subnet
- **`VQIPManager_UpdateRequest`** to update IPv6 subnet

`V6_SUBNET_REC` structure needs to be used in messages for all of above operations. Refer to Table 2-158, “V6_SUBNET_REC attributes” (p. 258) for the details of **`V6_SUBNET_REC`** structure.

Join V6 subnet

Overview

The northnound system will use **`VQIPManager_MergeRequest`** operation using MergeRequest to join two V6 subnets. Details of **`MergeRequest`** message are given in "Request structure - MergeRequest message" (p. 2-25).

Renumber V6 subnet

Overview

The northbound system will use **VQIPManager_RenumberRequest** operation using **RenumberRequest** to renumber IPv6 subnet and all the associated IP addresses and the corresponding DNS entries. Details of **RenumberRequest** message are given in the following table.

Table 2-157 RenumberRequest message structure

Part name	Type	Opt/ Req	Default	Description
commonInfo	**qipws:CommonInfo**	R	None	Contains user organization and locale. Please refer to Table 2-2, "CommonInfo parameters" (p. 7) for details about **CommonInfo** structure.
jobReq	**qipwx:JOB_REQUEST**	O	None	Data used to identify when to schedule the renumber request. Refer to "JOB_REQUEST" (p. 3-16) for details about the **JOB_REQUEST** structure. If this attribute is not specified, the request will be executed immediately.
source	**qipws:VQIP_BASE_ ENTIRY**	R	None	**V6_SUBNET_REC** identifying the source subnet to be renumbered using the startAddress attribute.
target	**qipws:VQIP_BASE_ ENTITY**	R	None	**V6_SUBNET_REC** identifying the destination subnet using the startAddress attribute.

Search V6 subnet

Overview

The northbound system invokes **`VQIPManager_PageSearchRequest`** operation to retrieve a list of subnets. The operation uses **`PageSearchRequest`** message with **`SEARCH_V6_SUBNET_REC`** structure including the search criteria. Details of **`V6_SUBNET_REC`** attributes are included in the following table. If no search criteria are specified, all the subnets are returned.

Table 2-158 V6_SUBNET_REC attributes

Part name	Type	Opt /Req	Description
`name`	**`xsd:string`**	O	Subnet Name
`startAddress`	**`xsd:string`**	O	Starting address of the subnet in IpV6 format, for example, 1234:5678::.
`subnetLength`	**`xsd:short`**	O	Length of the new subnet prefixes to be created. Valid values are 9-128.

Split V6 subnet

Overview

The northbound system uses **VQIPManager_SplitRequest** operation with **SplitRequest** message to split an existing subnet. Details of **SplitRequest** message are given in the following table.

Table 2-159 SplitRequest message attributes

Part name	Type	Opt/ Req	Default	Description
commonInfo	**qipws:CommonInfo**	R	None	Contains user organization and locale. Please refer to Table 2-2, "CommonInfo parameters" (p. 7) for details about **CommonInfo** structure.
jobReq	**qipws:JOB_REQUEST**	O	None	Data used to identify when to schedule the split request. Refer to "JOB_REQUEST" (p. 3-16) for details about the **JOB_REQUEST** structure. If this attribute is not specified, the request will be executed immediately.
entity	**qipws:VQIP_BASE_ ENTITY**	R	None	**V6_SUBNET_REC** identifying the subnet being split using the startAddress attribute.
requestedSize	**xsd:short**	R	None	The new subnet length.

3 Enumeration types

Overview

Purpose

This chapter contains information on the data structures used by the VitalQIP Web Service.

Contents

This chapter covers these topics.

Enumeration types and user defined objects

Overview

This section provides information on the data structures used by the operations supported in the VitalQIP Web Service interface.

AC_SUBSCRIBER_STATUS_TYPE

Overview

This is used in AcSubscribers to tag their operational status.

```
<!-- define enum AC_SUBSCRIBER_STATUS_TYPE -->
<xsd:simpleType name="AC_SUBSCRIBER_STATUS_TYPE">
 <xsd:restriction base="xsd:string">
    <xsd:enumeration value="ACTIVE"/>
      <xsd:enumeration value="INACTIVE"/>
    <xsd:enumeration value="PENDING"/>
 </xsd:restriction>
</xsd:simpleType>
```

ACTION_TYPE

Overview

This is used to specify different operations supported by VitalQIP Web Service. The schema for this data type is as shown below:

```
<!-- define enum ACTION_TYPE-->

<xsd:simpleType name="ACTION_TYPE">
<xsd:restriction base="xsd:string">
   <xsd:enumeration value ="A" />
   <xsd:enumeration value ="D" />
   <xsd:enumeration value ="S" />
   <xsd:enumeration value ="U" />
   <xsd:enumeration value ="G" />
   </xsd:restriction>
</xsd:simpleType>
```

Enumeration values are:

A	Add an object.
D	Delete an object.
S	Search for an object.
U	Update an object.
G	Update an object.

ADDRESS_BLOCK_KEYS

Overview

ADDRESS_BLOCK_KEY complex type is used to specify the address block object in operations requiring VQIP_KEY_ENTITY as input, such as, free pending block operation (VQIPManager_FreePendingBlockRequest).

```
<!--  Define Type ADDRESS_BLOCK_KEYS -->
<xsd:complexType name="ADDRESS_BLOCK_KEYS">
     <xsd:complexContent>
       <xsd:extension base="qipws:VQIP_KEY_ENTITY">
         <xsd:sequence>
           <xsd:element name="startAddress" type="xsd:string"/>
           <xsd:element name="prefixLength" type="xsd:short"/>
           <xsd:element name="addressType" type="qipws:V4V6_TYPE"
   minOccurs="0"/>
           <xsd:element name="pool" type="qipws:POOL_KEYS"
   minOccurs="0"/>
           <xsd:element name="blockStatus"
   type="qipws:BLOCK_STATUS_TYPE" minOccurs="0"/>
         </xsd:sequence>
       </xsd:extension>
     </xsd:complexContent>
     </XSD:COMPLEXTYPE>
```

ADDRESS_TYPE

Overview

This is used in templates to determine the type of address defined by the template. The values are self-explanatory.

```
<!-- define enum ADDRESS_TYPE -->
<xsd:simpleType name="ADDRESS_TYPE">
     <xsd:restriction base="xsd:string">
       <xsd:enumeration value="STATIC"/>
       <xsd:enumeration value="DYNAMIC"/>
       <xsd:enumeration value="RESERVED"/>
     </xsd:restriction>
</xsd:simpleType>
```

ALLOCATION_ALGORITHM_TYPE

Overview

This is used to specify different allocation algorithms supported by VitalQIP Web Service. One of these types need to specified as address allocation algorithm during Address Pool creation/modification.

```
<!-- define enum ALLOCATION_ALGORITHM_TYPE -->
<xsd:simpleType name="ALLOCATION_ALGORITHM_TYPE">
<xsd:restriction base="xsd:string">
   <xsd:enumeration value ="BEST_FIT_FROM_END" />
   <xsd:enumeration value ="BEST_FIT_FROM_START" />
   <xsd:enumeration value ="SPARSE_ALLOCATION" />
</xsd:restriction>
</xsd:simpleType>
```

BLOCK_STATUS_TYPE

Overview

This is used to specify status of Address Block.

```
<!-- define enum BLOCK_STATUS_TYPE -->
<xsd:simpleType name="BLOCK_STATUS_TYPE">
<xsd:restriction base="xsd:string">
   <xsd:enumeration value ="ALLOCATED" />
   <xsd:enumeration value ="FREE" />
   <xsd:enumeration value ="NEXT" />
   <xsd:enumeration value ="ORIGIN" />
   <xsd:enumeration value ="PENDING" />
   <xsd:enumeration value ="RESERVED" />
   <xsd:enumeration value ="SELECTED" />
   <xsd:enumeration value ="SITE" />
   <xsd:enumeration value ="USED" />
</xsd:restriction>
</xsd:simpleType>
```

ALLOCATED

Block has been givepn to a pool.

FREE

Block is eligible for allocation.

NEXT

Block is the next one to be allocated according to SPL algorithm. This status is applicable for Sparse Allocation (SPL) algorithm only.

ORIGIN

Block has allocated into child blocks within the current pool.

PENDING

Block is in transition from “USED” to “FREE” status.

RESERVED

Block is not available for allocation. The block is kept in this state as a result of an explicit action by the user.

SELECTED

This is a transient state for very short time.

SITE

The single address block obtained during allocation will be sub-allocated further to a specified pool and the subnets corresponding to these sub-blocks will be created in VitalQIP.

USED

Block has been assigned to a subnet/organization.

BLOCK_STEP_TYPE

Overview

BLOCK_STEP_TYPE is used to define three steps involved in block allocation process. If one or more processing steps of block operation fails, then this define is used to identify the step(s) that failed.

```
<!-- define enum BLOCK_STEP_TYPE -->
<xsd:simpleType name="BLOCK_STEP_TYPE">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="ALLOCATE"/>
    <xsd:enumeration value="CREATE_INFRASTRUCTURE"/>
    <xsd:enumeration value="RIR_REPORT"/>
  </xsd:restriction>
</xsd:simpleType>
```

ALLOCATE

First of three steps in block allocation process. Step that actually allocate the block to the child pool.

CREATE_INFRASTRUCTURE

Second of three steps in block allocation process. Step to create infrastructure objects in VitalQIP for a block after it has been allocated.

RIR_REPORT

Third of three steps in block allocation process. Step to Generate RIR Report, if required.

FREE_PENDING_BLOCK_TYPE

Overview

This is used to specify one of the supported options to indicate when pending block is changed from “PENDING” to “FREE” status.

```
<!-- define enum FREE_PENDING_BLOCK_TYPE -->
<xsd:simpleType name="FREE_PENDING_BLOCK_TYPE">
<xsd:restriction base="xsd:string">
   <xsd:enumeration value ="FREE_AFTER_DAYS" />
   <xsd:enumeration value ="FREE_AFTER_DAYS_AND_WHOIS" />
   <xsd:enumeration value ="FREE_AFTER_DAYS_OR_WHOIS" />
   <xsd:enumeration value ="FREE_AFTER_WHOIS" />
</xsd:restriction>
</xsd:simpleType>
```

FREE_AFTER_DAYS

Change status after specified number of days.

FREE_AFTER_DAYS_AND_WHOIS

hange status after both of following criteria are met:

- After specified number of days
- After the block is removed from WHOIS database

FREE_AFTER_DAYS_OR_WHOIS

Change status after either of following criteria are met:

- After specified number of days

OR

- After the block is removed from WHOIS database

FREE_AFTER_WHOIS

Change status after the block is removed from WHOIS database.

FREE_PENDING_CHECK_TYPE

Overview

This is used to specify the level at which to check and free the pending blocks for checkPendingBlock Request.

```
<!-- define enum FREE_PENDING_CHECK_TYPE -->
<xsd:simpleType name="FREE_PENDING_CHECK_TYPE">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="BLOCK"/>
    <xsd:enumeration value="POOL"/>
    <xsd:enumeration value="ORGANIZATION"/>
    <xsd:enumeration value="ALL"/>
  </xsd:restriction>
</xsd:simpleType>
```

BLOCK

Check and Free a specified pending block.

POOL

Check and Free pending blocks per pool. All pending blocks for the specified pool are freed.

ORGANIZATION

Free pending blocks per organization. All pending blocks for the specified organization are freed.

ALL

Free all pending blocks.

INTERFACE_KEYS

Overview

Used to identify an interface. For example, while adding an ipaddress to an interface.

```
<xsd:complexType name="INTERFACE_KEYS">
 <xsd:complexContent>
  <xsd:extension base="qipws:VQIP_KEY_ENTITY">
    <xsd:sequence>
      <xsd:element name="name" type="xsd:string"/>
      <xsd:element name="nodeKey" type="qipws:NODE_KEYS"/>
    </xsd:sequence>
  </xsd:extension>
 </xsd:complexContent>
</xsd:complexType>
```

IPADDR_KEYS

Overview

Used to identify an IP address. For example, while adding a domain information record to an IP Address.

```
<xsd:complexType name="IPADDR_KEYS">
 <xsd:complexContent>
  <xsd:extension base="qipws:VQIP_KEY_ENTITY">
    <xsd:sequence>
      <xsd:element name="ipAddrStr" type="xsd:string"/>
      <xsd:element name="interfaceKey"
  type="qipws:INTERFACE_KEYS" minOccurs="0"/>
      <xsd:element name="subnetStrtAddr"
  type="qipws:V6_SUBNET_KEYS" minOccurs="0"/>
    </xsd:sequence>
</xsd:extension>
 </xsd:complexContent>
</xsd:complexType>
```

IP_ADDR_ALLOC_ALG_TYPE

Overview

Used when IP addresses are created within subnets. This is one of the parameters that is specified when subnets are created, but is used when IP addresses are created within subnets. **EXISTING** is not used in the context of subnet creation.

```
<!-- define enum IPADDR_ALLOC_ALG_TYPE  -->
<xsd:simpleType name="IPADDR_ALLOC_ALG_TYPE">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="NEXT_AVAILABLE"/>
    <xsd:enumeration value="LAST_AVAILABLE"/>
    <xsd:enumeration value="RANDOM"/>
    <xsd:enumeration value="SPECIFIC"/>
    <xsd:enumeration value="EXISTING"/>
  </xsd:restriction>
</xsd:simpleType>
```

JOB_REQUEST

Overview

Used to schedule an operation.

```
<!-- Define type JOB_REQUEST -->
<xsd:complexType name="JOB_REQUEST">
<xsd:sequence>
   <xsd:element name="when" type="xsd:string" minOccurs="0"/>
   <xsd:element name="toEmail" type="xsd:string" minOccurs="0"/>
   <xsd:element name="ccEmail" type="xsd:string" minOccurs="0"/>
   <xsd:element name="sendEmail" type="xsd:boolean"
   minOccurs="0"/>
   <xsd:element name="runMode" type="qipws:SCHEDULE_JOB_TYPE"
   minOccurs="0"/>
   <xsd:element name="description" type="xsd:string"
   minOccurs="0"/>
</xsd:sequence>
</xsd:complexType>
```

when

Time when the job will be started. The format of the date can be any of the following formats :

- MM/dd/yyyy hh:mm
- yyyy-MM-dd hh:mm
- yyyy-MM-dd hh:mm z
- yyyyMMdd'T'hhmmZ
- yyyy-MM-dd h:mma
- MMM d HH:mm yyyy
- EEE MMM d HH:mm:ss zzz yyyy

toEmail

Indicates the email addresses to which to send email when the job completes. Addresses are comma separated.

ccMail

Indicates the email addresses to which to carbon copy the job email when the job completes. Addresses are comma separated.

sendMail

Indicates that the job should send email. If the **`toEmail`** argument is not specified, by default the email goes to the email address specified in the administrator profile. If no address is specified in the administrator profile, an error is displayed.

runMode

Of type SCHEDULE_JOB_TYPE - determines when the request will be executed.

description

Optional description of the job.

LIST_TYPE

Overview

This enumerated list is used to specify the object type for which all the values for a “key” field, such as **name**, will be retrieved. The values described below are self-explanatory. **OBJ_CLASS** should be used for retrieving node types.

```
<!-- define enum LIST_TYPE - -->
<xsd:simpleType name="LIST_TYPE">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="DHCP_SRV"/>
    <xsd:enumeration value="DHCP_OPT_TEMPL"/>
    <xsd:enumeration value="DHCP_SCOPE_POL_TEMPL"/>
    <xsd:enumeration value="DNS_SRV"/>
    <xsd:enumeration value="DOMAIN"/>
    <xsd:enumeration value="OBJ_CLASS"/>
    <xsd:enumeration value="SUBNET_ORGANIZATION"/>
    <xsd:enumeration value="USER_CLASS"/>
    <xsd:enumeration value="VENDOR_CLASS"/>
  </xsd:restriction>
</xsd:simpleType>
```

NAMEVALUE

Overview

This is used to specify a name/value pair. The schema for this data type is as shown below.

This structure is included to allow for attributes that may need to be added later and also to allow definition and use of User Defined Fields (UDFs).

```
<!-- define struct NameValue-->

<complexType name="NameValue">
<xsd:sequence>
   <xsd:element name="name" type="xsd:string" />
   <xsd:element name="value" type="xsd:string" minOccurs="0î/>
</xsd:sequence>
</complexType>
```

name

Name of the entity of string type, for example, an attribute name.

value

Value of the entity of string type, for example, an attribute value.

NODE_KEYS

Overview

Used to identify a node, for example, when an interface is added to a node.

```
<xsd:complexType name="NODE_KEYS">
 <xsd:complexContent>
  <xsd:extension base="qipws:VQIP_KEY_ENTITY">
    <xsd:sequence>
      <xsd:element name="uniqueID" type="xsd:string"/>
    </xsd:sequence>
  </xsd:extension>
 </xsd:complexContent>
</xsd:complexType>
```

PARTIAL_FAILURE_REC

Overview

This structure is used in the response of an operation involving multiple processing steps and status of each of the steps need to be reported.

```
<!-- Define Type PARTIAL_FAILURE_REC -->
<xsd:complexType name="PARTIAL_FAILURE_REC">
  <xsd:sequence>
    <xsd:element name="step" type="qipws:BLOCK_STEP_TYPE"/>
    <xsd:element name="failureMessage" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>
```

step

Indicates one of three steps that failed (ALLOCATE, CREATE_INFRASTRUCTURE or RIR_REPORT).

failureMessage

Contains the cause of the failure of the corresponding step.

POOL_KEYS

Overview

This structure is used to identify a pool.

```
<!-- Define Type POOL_KEYS -->
<xsd:complexType name="POOL_KEYS">
  <xsd:complexContent>
    <xsd:extension base="qipws:VQIP_KEY_ENTITY">
      <xsd:sequence>
        <xsd:element name="name" type="xsd:string"/>
        <xsd:element name="parentName" type="xsd:string"
  minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

name

Name of the pool.

parentName

Name of the parent pool. Not applicable for a seed pool.

REGISTRY_TYPE

Overview

This is used to specify the source of IP address block for pool.

```
<!-- define enum REGISTRY_TYPE -->
<xsd:simpleType name="REGISTRY_TYPE">
<xsd:restriction base="xsd:string">
  <xsd:enumeration value ="AFRINIC" />
  <xsd:enumeration value ="APNIC" />
  <xsd:enumeration value ="ARIN" />
  <xsd:enumeration value ="RIPE" />
</xsd:restriction>
</xsd:simpleType>
```

AFRINIC

Indicates that the source of address space for the pool is AfriNIC and VitalQIP should send inetnum reports to AfriNIC as the address space is assigned.

APNIC

Indicates that the source of address space for the pool is APNIC and VitalQIP should send inetnum reports to APNIC as the address space is assigned.

ARIN

Indicates that the source of address space for the pool is ARIN and VitalQIP should send SWIP reports to ARIN as the address space is assigned.

RIPE

Indicates that the source of address space for the pool is RIPE and VitalQIP should send inetnum reports to RIPE as the address space is assigned.

REGISTRATION_ACTION_TYPE

Overview

This enumeration is used to define allowable values for Registration Action attribute in ARIN-Reassign-Simple template.

```
<!-- define enum REGISTATION_ACTION_TYPE-->
<xsd:simpleType name="REGISTRATION_ACTION_TYPE">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="NEW"/>
    <xsd:enumeration value="MODIFY"/>
    <xsd:enumeration value="REMOVE"/>
  </xsd:restriction>
</xsd:simpleType>
```

RESPONSE_DATA

Overview

Data returned along with the response of a scheduled job.

```
<!-- Define type RESPONSE_DATA -->
<xsd:complexType name="RESPONSE_DATA">
  <xsd:sequence>
    <xsd:element name="jobId" type="xsd:int" minOccurs="0"/>
    <xsd:element name="name" type="xsd:string" minOccurs="0"/>
    <xsd:element name="completionTime" type="xsd:string"
  minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>
```

jobid

Unique identifier of the scheduled job.

name

The name of the scheduled operation. For example, when adding V6 subnets, the value of **name** is **ADD_V6_SUBNETS**.

completionTime

Time when the job was completed (if applicable).

RESULT_TYPE

Overview

This is used to specify various success statuses of the operations. The schema for this data type is as shown below:

```
<!-- define enum RESULT_TYPE-->

<!-- define enum RESULT_TYPE -->
<xsd:simpleType name="RESULT_TYPE">
     <xsd:restriction base="xsd:string">
          <xsd:enumeration value="SUCCESS"/>
          <!-- NOT_FOUND applicable for delete operation -->
          <xsd:enumeration value="NOT_FOUND"/>
          <xsd:enumeration value="PARTIAL_FAILURE"/>
          <xsd:enumeration value="UNKNOWN"/>
     </xsd:restriction>
</xsd:simpleType>
```

SUCCESS

All operations were successful in VitalQIP.

NOT_FOUND

Applicable only for Get operation. Indicates that the record was not found in the VitalQIP database.

PARTIAL_FAILURE

Applicable only for operations that involve multiple processing steps and status of each of the steps needs to be reported

RIR_REPORT_TYPE

Overview

This is used to specify a report for RIR Reporting. The schema for this data type is as shown below:

```
<!-- define enum RIR_REPORT_TYPE -->
<xsd:simpleType name="RIR_REPORT_TYPE">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="ARIN-Reallocate"/>
    <xsd:enumeration value="ARIN-Reassign-Detailed"/>
    <xsd:enumeration value="ARIN-Reassign-Simple"/>
    <xsd:enumeration value="ARIN-IPv6-Reallocate"/>
    <xsd:enumeration value="ARIN-IPv6-Reassign"/>
    <xsd:enumeration value="INETNUM for IPv4"/>
    <xsd:enumeration value="INETNUM for IPv6"/>
    <xsd:enumeration value="Do Not Generate Report"/>
  </xsd:restriction>
</xsd:simpleType>
```

Enumeration values are defined following:

ARIN-Reallocate

ARIN-Reallocated-4.0 SWIP template for IPv4 address blocks.

ARIN-Reassign-Detailed

ARIN-Reassign-Detailed-4.0 SWIP template for IPv4 address blocks.

ARIN-Reassign-Simple

ARIN-Reassign-Simple-3.2.1 SWIP template for IPv4 address blocks.

ARIN-IPV6-Reallocate

ARIN-IPV6-Reallocate-3.2.3 SWIP template for IPv6 address blocks.

ARIN-IPV6-Reassign

ARIN-IPV6-Reassign-3.2.3 SWIP template for IPv6 address blocks.

INETNUM for IPV4

INETNUM for IPV4 Report for RIPE/APNIC/AFRINIC.

INETNUM for IPV6

INETNUM for IPV6 Report for RIPE/APNIC/AFRINIC.

Do Not Generate Report

Do Not Generate Report.

RULE_COMPONENT_TYPE

Overview

This is an enumeration type that is used to specify the type of rule component. The schema of **RULE_COMPONENT_TYPE** is given below:

```
<!-- define enum RULE_COMPONENT_TYPE -->
  <xsd:simpleType name="RULE_COMPONENT_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="USED"/>
      <xsd:enumeration value="RESERVED"/>
    </xsd:restriction>
  </xsd:simpleType>
```

Used

Indicates when a block is allocated using this rule, the block should be given a Block Status of Used and a corresponding subnet should be created in VitalQIP.

Reserved

Indicates when a block is allocated using this rule, the block should be given a Block Status of Reserved. Until the block status is changed to Free by an explicit action, this address block is not used for any allocation activity.

RULE_LEVEL

Overview

This is used to specify the level up to which the administrators can specify rules. The schema for this data type is as shown below:

```
<!-- define enum RULE_LEVEL-->

<xsd:simpleType name="RULE_LEVEL">
<xsd:restriction base="xsd:string">
   <xsd:enumeration value ="NORMAL" />
   <xsd:enumeration value ="ADVANCED" />
   <xsd:enumeration value ="EXPERT" />
</xsd:restriction>
</xsd:simpleType>
```

Normal

Rules can be processed only by administrators who have the ability to allocate blocks can process rules, that is, those that are specified as Normal in the VitalQIP Administrator profile.

Advanced

Master, Organization and Normal administrators can only process rules.

Expert

Only Master, Organization, Normal and Expert administrators can process rules.

RULE_TYPE

Overview

This is used to specify the type of rule. The schema for this data type is as shown below:

```
<!-- define enum RULE_TYPE -->

<xsd:simpleType name="RULE_TYPE">
<xsd:restriction base="xsd:string">
   <xsd:enumeration value ="Used" />
   <xsd:enumeration value ="FREE" />
   <xsd:enumeration value ="RESERVED" />
   <xsd:enumeration value ="SITE" />
</xsd:restriction>
</xsd:simpleType>
```

Used

Indicates when a block is allocated using this rule, the block should be given a block status of Used and a corresponding subnet should be created in VitalQIP.

Free

Indicates when a block is allocated using this rule, the block should be given a block status of Free. No other action will be taken during allocation.

Reserved

Indicates when a block is allocated using this rule, the block should be given a block status of Reserved. Until the block status is changed to Free by an explicit action, this address block is not used for any allocation activity.

Site

The single address block obtained during allocation will be sub-allocated further to a specified pool and the subnets corresponding to these sub-blocks will be created in VitalQIP.

SCHEDULE_JOB_TYPE

Overview

Used to determine when a job should be scheduled. Included in JOB_REQUEST object.

```
<!-- define enum SCHEDULE_JOB_TYPE -->
<xsd:simpleType name="SCHEDULE_JOB_TYPE">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="IN_FOREGROUND"/>
    <xsd:enumeration value="IN_BACKGROUND"/>
    <xsd:enumeration value="SCHEDULED_AT"/>
  </xsd:restriction>
</xsd:simpleType>
```

IN_FOREGROUND

Schedule job immediately, in fore ground. As expected, setting this attribute results in a synchronous request, resulting in the north bound system waiting for the job to be completed.

IN_BACKGROUND

Schedule job immediately, in back ground. As expected, setting this attribute results in an asynchronous request and does not result in the north bound system waiting for the job to be completed.

SCHEDULED_AT

Schedule job at later specified time, in back ground. The time is specified using the when attribute of JOB_REQUEST object. As expected, setting this attribute results in an asynchronous request and does not result in the north bound system waiting for the job to be completed.

SERVER_KEY

Overview

SERVER_KEY complex type is used to specify the server object being sent in by the northbound interface. For example, during Clone server operation.

```
<!-- define struct SERVER_KEY -->
<xsd:complexType name="SERVER_KEY">
  <xsd:complexContent>
    <xsd:extension base="qipws:VQIP_KEY_ENTITY">
      <xsd:sequence>
        <xsd:element name="host" type="xsd:string"/>
        <xsd:element name="domain" type="xsd:string"/>
        <xsd:element name="serverType"
type="qipws:SERVER_TYPE"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
  </XSD:COMPLEXTYPE>
```

SUBNET_ALLOCATION_TYPE

Overview

Used while creating subnets.

```
<!-- define enum SUBNET_ALLOCATION_TYPE -->
<xsd:simpleType name="SUBNET_ALLOCATION_TYPE">
  <xsd:restriction base="xsd:string">
       <xsd:enumeration value="AUTOMATIC"/>
       <xsd:enumeration value="EXPLICIT"/>
  </xsd:restriction>
</xsd:simpleType>
```

AUTOMATIC

Subnet(s) will be created under the Default Seed Pool using BFS algorithm. Subnet addresses will be generated in a contiguous fashion. But if the contiguous space is not available, the next best address that is available will be used.

EXPLICIT

The subnet addresses will be generated sequentially using the value of START_TYPE to determine whether to start at this prefix or next. If any of these subnets cannot be created in a contiguous fashion due to space not being available, the entire request will be rejected.

SUBNET_CREATION_OPTION_TYPE

Overview

Valid options for subnet creation during explicit block allocation:

```
<!-- define enum SUBNET_CREATION_OPTION_TYPE -->
<xsd:simpleType name="SUBNET_CREATION_OPTION_TYPE">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="CREATE_NEW"/>
    <xsd:enumeration value="LINK_TO_EXISTING"/>
    <xsd:enumeration value="CREATE_NEW_IF_NECESSARY"/>
    <xsd:enumeration value="DO_NOT_CREATE_OR_LINK"/>
  </xsd:restriction>
</xsd:simpleType>
```

CREATE_NEW

Indicates to create a subnet at the address and size of the block. If subnet already exists, the allocation process fails.

LINK_TO_EXISTING

Indicates to link to existing subnet. There must be a subnet in VitalQIP at the address and size of the allocated block. If such a block does not exist, then the allocation request will fail.

CREATE_NEW_IF_NECESSARY

Indicates to create a subnet if it already exists. If it already exists, the block is linked with the subnet. If a subnet does not exist, it is created.

DO_NOT_CREATE_OR_LINK

Indicates that subnet should not be created or linked to in Address Management.

START_TYPE

Overview

Used for creating subnets where **SUBNET_ALLOCATION_TYPE** is EXPLICIT.

```
<!-- define enum START_TYPE -->
  <xsd:simpleType name="START_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="AT"/>
      <xsd:enumeration value="AFTER"/>
    </xsd:restriction>
  </xsd:simpleType>
```

AT

Create subnets at the specified prefix.

AFTER

Create the subnets after the specified prefix.

UDA_DEFINITION_ATTRIBUTE_TYPE

Overview

The definition of a UDA includes the UDA_DEFINITION_ATTRIBUTE_TYPE. This declares the UDA value's datatype.

```
<xsd:simpleType name="UDA_DEFINITION_ATTRIBUTE_TYPE">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="BOOLEAN"/>
    <xsd:enumeration value="IPV4_ADDRESS"/>
    <xsd:enumeration value="IPV6_ADDRESS"/>
    <xsd:enumeration value="MULTILINED_TEXT"/>
    <xsd:enumeration value="NUMERIC"/>
    <xsd:enumeration value="LIST"/>
    <xsd:enumeration value="TEXT"/>
  </xsd:restriction>
  </xsd:simpletype>
```

UDA_DEFINITION_INFRASTRUCTURE_TYPE

Defines a UDA's infrastructure type association.

```
<xsd:simpleType name="UDA_INFRASTRUCTURE_TYPE">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="AC_DEVICE"/>
    <xsd:enumeration value="AC_SUBSCRIBER"/>
    <xsd:enumeration value="BLOCK"/>
    <xsd:enumeration value="DC_SERVER"/>
    <xsd:enumeration value="DHCP_SERVER"/>
    <xsd:enumeration value="DNS_SERVER"/>
    <xsd:enumeration value="DNS_VIEW"/>
    <xsd:enumeration value="INTERFACE"/>
    <xsd:enumeration value="IPV4_ADDRESS"/>
    <xsd:enumeration value="IPV6_ADDRESS"/>
    <xsd:enumeration value="IPV4_SUBNET"/>
    <xsd:enumeration value="IPV6_SUBNET"/>
    <xsd:enumeration value="MANAGED_FILE"/>
    <xsd:enumeration value="MYVIEW"/>
    <xsd:enumeration value="NETWORK"/>
    <xsd:enumeration value="NODE"/>
    <xsd:enumeration value="ORGANIZATION"/>
    <xsd:enumeration value="POOL"/>
    <xsd:enumeration value="SUBNET_ORGANIZATION"/>
    <xsd:enumeration value="ZONE"/>
  </xsd:restriction>
  </xsd:simpletype>
```

UDAGROUP

Overview

This object provides a logical grouping of User Defined Attributes. For example, User Defined Attributes **`City`** and **`State`** may be included in **`UdaGroup`** Address.

```
<xsd:complexType name="UdaGroup">
     <xsd:sequence>
       <xsd:element name="name" type="xsd:string"/>
       <xsd:element name="udas" type="qipws:NameValue"
  maxOccurs="unbounded"/>
     </xsd:sequence>
  </xsd:complexType>
```

NAME

Name of the group.

udas

List of UDAs associated with this group.

UDALIST

Overview

List of User Defined Attributes and the groups to which they belong.

```
<xsd:complexType name="UdaList">
     <xsd:sequence>
        <xsd:element name="udas" type="qipws:NameValue"
   minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="groups" type="qipws:UdaGroup"
   minOccurs="0" maxOccurs="unbounded"/>
     </xsd:sequence>
   </xsd:complexType>
```

udas

List of one more objects that include the name of the User Defined Attribute and the value associated with the User Defined Attribute.

groups

List of UDA groups.

V4V6_TYPE

Overview

Identifies a rule as IPv4 or IPv6.

```
<xsd:simpleType name="V4V6_TYPE">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="IPv4"/>
    <xsd:enumeration value="IPv6"/>
  </xsd:restriction>
</xsd:simpleType>
```

IPv4

Indicates that the rule is an IPv4 rule.

IPv6

Indicates that the rule is an IPv6 rule.

V6_SUBNET_KEYS

Overview

Operations that require V6 Subnets as input, such as V6 Subnet Add/Modify/Delete/Update, will use the V6_SUBNET_KEYS complex type.

```
<xsd:complexType name="V6_SUBNET_KEYS">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="startAddress" type="xsd:string"/>
          <xsd:element name="prefixLength" type="xsd:short"
 minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
```

4 Sample Request and Response messages

Overview

Purpose

This chapter provides sample request and response messages for each supported operation.

Contents

This chapter covers these topics.

Add operation

Request

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
 </soapenv:Header>
  <soapenv:Body>
    <ns1:AddRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:AddRequest">
       <ns1:commonParam>
         <ns1:organization>VitalQIP Organization</ns1:organization>
         <ns1:locale>en_US</ns1:locale>
       </ns1:commonParam>
       <ns1:reqObject xsi:type="ns1:RULE_REC">
         <ns1:name>NEW_QIP_V6_SIZE_40_USED</ns1:name>
         <ns1:ruleLevel>NORMAL</ns1:ruleLevel>
         <ns1:ruleType>USED</ns1:ruleType>
         <ns1:addrTemplate>IPV6_ADDR_TMPL_8</ns1:addrTemplate>
         <ns1:requestedSize>40</ns1:requestedSize>
         <ns1:v4V6Rule>IPv6</ns1:v4V6Rule>
       </ns1:reqObject>
    </ns1:AddRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
       <ns1:result>SUCCESS</ns1:result>
    </ns1:Response>
  </soapenv:Body>
</soapenv:Envelope>
```

Delete operation

Request

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:DeleteRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:DeleteRequest">
       <ns1:commonParam>
         <ns1:organization>VitalQIP Organization</ns1:organization>
         <ns1:locale>en_US</ns1:locale>
       </ns1:commonParam>
       <ns1:reqObject xsi:type="ns1:RULE_REC">
         <ns1:name>NEW_QIP_V6_SIZE_40_USED</ns1:name>
       </ns1:reqObject>
    </ns1:DeleteRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
       <ns1:result>SUCCESS</ns1:result>
    </ns1:Response>
  </soapenv:Body>
</soapenv:Envelope>
```

Get operation

Request

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:GetRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:GetRequest">
       <ns1:commonParam>
         <ns1:organization>VitalQIP Organization</ns1:organization>
         <ns1:locale>en_US</ns1:locale>
       </ns1:commonParam>
       <ns1:reqObject xsi:type="ns1:RULE_REC">
         <ns1:name>QIP_V6_SIZE_40_USED</ns1:name>
       </ns1:reqObject>
    </ns1:GetRequest>
```

Get Response

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:GetResponse xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:GetResponse">
       <ns1:result>SUCCESS</ns1:result>
       <ns1:record xsi:type="ns1:RULE_REC">
         <ns1:name>QIP_V6_SIZE_40_USED</ns1:name>
         <ns1:ruleLevel>NORMAL</ns1:ruleLevel>
         <ns1:ruleType>USED</ns1:ruleType>
         <ns1:addrTemplate>IPV6_ADDR_TMPL_8</ns1:addrTemplate>
         <ns1:requestedSize>40</ns1:requestedSize>
         <ns1:v4V6Rule>IPv6</ns1:v4V6Rule>
       </ns1:record>
    </ns1:GetResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

PageSearch operation

Request

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:PageSearchRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:PageSearchRequest">
       <ns1:commonParam>
         <ns1:organization>VitalQIP Organization</ns1:organization>
         <ns1:locale>en_US</ns1:locale>
       </ns1:commonParam>
       <ns1:reqObject xsi:type="ns1:SEARCH_POOL_REC">
         <ns1:name>QIP_CHILD_POOL_1_1</ns1:name>
       </ns1:reqObject>
       <ns1:pageCount>1</ns1:pageCount>
       <ns1:pageSize>0</ns1:pageSize>
    </ns1:PageSearchRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:PageSearchResponse xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:PageSearchResponse">
       <ns1:result>SUCCESS</ns1:result>
       <ns1:objectData xsi:type="ns1:CHILD_POOL_REC">
         <ns1:name>QIP_CHILD_POOL_1_1</ns1:name>
         <ns1:parentName>QIP_SEED_POOL_1</ns1:parentName>
         <ns1:contact>satishjm@lucent.com</ns1:contact>
         <ns1:childAllocationNotify>false</ns1:childAllocationNotify>
         <ns1:v4Algorithm>BEST_FIT_FROM_START</ns1:v4Algorithm>
         <ns1:v4MaxAllocation>0</ns1:v4MaxAllocation>
         <ns1:v6Algorithm>BEST_FIT_FROM_START</ns1:v6Algorithm>
         <ns1:v6MaxAllocation>0</ns1:v6MaxAllocation>
         <ns1:recurseRequest>true</ns1:recurseRequest>
         <ns1:recurseRequestNotify>false</ns1:recurseRequestNotify>
```

```
        <ns1:v4Rule>SJM_V4_16_FREE_RULE</ns1:v4Rule>
        <ns1:v6Rule>QIP_V6_SIZE_40_USED</ns1:v6Rule>
      </ns1:objectData>
      <ns1:totalPages>1</ns1:totalPages>
      <ns1:currentPage>1</ns1:currentPage>
    </ns1:PageSearchResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Search operation

Request

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:SearchRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:SearchRequest">
       <ns1:commonParam>
         <ns1:organization>VitalQIP Organization</ns1:organization>
         <ns1:locale>en_US</ns1:locale>
       </ns1:commonParam>
       <ns1:reqObject xsi:type="ns1:RULE_REC">
         <ns1:name>*</ns1:name>
       </ns1:reqObject>
    </ns1:SearchRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:SearchResponse xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:SearchResponse">
       <ns1:result>SUCCESS</ns1:result>
       <ns1:objectData xsi:type="ns1:RULE_REC">
         <ns1:name>SJM_V4_16_FREE_RULE</ns1:name>
         <ns1:ruleLevel>NORMAL</ns1:ruleLevel>
         <ns1:ruleType>FREE</ns1:ruleType>
         <ns1:requestedSize>16</ns1:requestedSize>
         <ns1:v4V6Rule>IPv4</ns1:v4V6Rule>
       </ns1:objectData>
       <ns1:objectData xsi:type="ns1:RULE_REC">
         <ns1:name>bala_254</ns1:name>
         <ns1:ruleLevel>NORMAL</ns1:ruleLevel>
         <ns1:ruleType>USED</ns1:ruleType>
         <ns1:addrTemplate>bala_254</ns1:addrTemplate>
```

```
        <ns1:requestedSize>24</ns1:requestedSize>
        <ns1:v4V6Rule>IPv4</ns1:v4V6Rule>
      </ns1:objectData>
      <ns1:objectData xsi:type="ns1:RULE_REC">
        <ns1:name>bala_26</ns1:name>
        <ns1:ruleLevel>NORMAL</ns1:ruleLevel>
        <ns1:ruleType>USED</ns1:ruleType>
        <ns1:addrTemplate>bala_26</ns1:addrTemplate>
        <ns1:requestedSize>26</ns1:requestedSize>
        <ns1:v4V6Rule>IPv4</ns1:v4V6Rule>
      </ns1:objectData>
      <ns1:objectData xsi:type="ns1:RULE_REC">
        <ns1:name>QIP_V4_SIZE_20_USED</ns1:name>
        <ns1:ruleLevel>NORMAL</ns1:ruleLevel>
        <ns1:ruleType>USED</ns1:ruleType>
        <ns1:addrTemplate>IPV4_PC_ROUTER_SIZE_8</ns1:addrTemplate>

   <ns1:subnetProfTemplate>QIP_SUBNET_PROFILE_TMPL</ns1:subnetProfTemplate>
        <ns1:requestedSize>20</ns1:requestedSize>
        <ns1:v4V6Rule>IPv4</ns1:v4V6Rule>
      </ns1:objectData>
      <ns1:objectData xsi:type="ns1:RULE_REC">
        <ns1:name>STARGÅTE</ns1:name>
        <ns1:ruleLevel>NORMAL</ns1:ruleLevel>
        <ns1:ruleType>USED</ns1:ruleType>
        <ns1:description>More stuff</ns1:description>
        <ns1:requestedSize>8</ns1:requestedSize>
        <ns1:v4V6Rule>IPv4</ns1:v4V6Rule>
      </ns1:objectData>
      <ns1:objectData xsi:type="ns1:RULE_REC">
        <ns1:name>QIP_V6_SIZE_40_USED</ns1:name>
        <ns1:ruleLevel>NORMAL</ns1:ruleLevel>
        <ns1:ruleType>USED</ns1:ruleType>
        <ns1:addrTemplate>IPV6_ADDR_TMPL_8</ns1:addrTemplate>
        <ns1:requestedSize>40</ns1:requestedSize>
        <ns1:v4V6Rule>IPv6</ns1:v4V6Rule>
      </ns1:objectData>
    </ns1:SearchResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Update operation

Request

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:UpdateRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:UpdateRequest">
      <ns1:commonParam>
        <ns1:organization>VitalQIP Organization</ns1:organization>
        <ns1:locale>en_US</ns1:locale>
      </ns1:commonParam>
      <ns1:reqObject xsi:type="ns1:RULE_REC">
        <ns1:name>UPDATED_QIP_V6_SIZE_40_USED</ns1:name>
        <ns1:ruleLevel>NORMAL</ns1:ruleLevel>
        <ns1:ruleType>USED</ns1:ruleType>
        <ns1:addrTemplate>IPV6_ADDR_TMPL_8</ns1:addrTemplate>
        <ns1:requestedSize>40</ns1:requestedSize>
        <ns1:v4V6Rule>IPv6</ns1:v4V6Rule>
      </ns1:reqObject>
      <ns1:key xsi:type="ns1:RULE_KEYS">
        <ns1:name>NEW_QIP_V6_SIZE_40_USED</ns1:name>
      </ns1:key>
    </ns1:UpdateRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
      <ns1:result>SUCCESS</ns1:result>
      <ns1:responseData />
    </ns1:Response>
  </soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_AddContainedObjectRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:AddContainedObjectRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:AddContainedObjectRequest">
      <ns1:commonParam>
        <ns1:organization>VitalQIP Organization</ns1:organization>
        <ns1:locale>en_US</ns1:locale>
      </ns1:commonParam>
      <ns1:parentKey xsi:type="ns1:NODE_KEYS">
        <ns1:uniqueID>node1</ns1:uniqueID>
      </ns1:parentKey>
      <ns1:containedObject xsi:type="ns1:INTERFACE_REC">
        <ns1:name>interface3</ns1:name>
        <ns1:macAddress>C1</ns1:macAddress>
        <ns1:ipAddrs xsi:type="ns1:IPADDR_REC">

   <ns1:ipAddrStr>1234:5678:0000:0000:0000:0000:0003:0001</ns1:ipAddrStr>
          <ns1:allocationAlgorithm>SPECIFIC</ns1:allocationAlgorithm>

   <ns1:subnetStrtAddr>1234:5678:0000:0000:0000:0000:0000:0000</ns1:subnetSt
   rtAddr>
        </ns1:ipAddrs>
      </ns1:containedObject>
    </ns1:AddContainedObjectRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_AddSeedBlockToSeedPoolrequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:AddSeedBlockToSeedPoolRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:AddSeedBlockToSeedPoolRequest">
      <ns1:commonParam>
        <ns1:organization>VitalQIP Organization</ns1:organization>
        <ns1:locale>en_US</ns1:locale>
      </ns1:commonParam>
      <ns1:addressBlock xsi:type="ns1:SEED_BLOCK_REC">
        <ns1:startAddress>192.168.1.0</ns1:startAddress>
        <ns1:prefixLength>24</ns1:prefixLength>
        <ns1:pool xsi:type="ns1:POOL_KEYS">
          <ns1:name>seedPool-A</ns1:name>
        </ns1:pool>
        <ns1:maintainer>myMaintainer</ns1:maintainer>
        <ns1:threshold>90</ns1:threshold>
        <ns1:reportTypes>ARIN-Reallocate</ns1:reportTypes>
        <ns1:reportTypes>ARIN-Reassign-Detailed</ns1:reportTypes>
        <ns1:reportTypes>ARIN-Reassign-Simple</ns1:reportTypes>
        <ns1:reportTypes>Do Not Generate Report</ns1:reportTypes>
      </ns1:addressBlock>
    </ns1:AddSeedBlockToSeedPoolRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
      <ns1:result>SUCCESS</ns1:result>
      <ns1:responseData />
    </ns1:Response>
  </soapenv:Body>
```

```
</soapenv:Envelope>
```

VQIPManager_AllocateBlockToPoolRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
  details.
</soapenv:Header>
<soapenv:Body>
         <ns1:AllocateBlockToPoolMessageRequest xmlns:ns1="http://alcatel-
  lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
  instance" xsi:type="ns1:AllocateBlockToPoolMessageRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization
       </ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:parameters>
               <ns1:pool xsi:type="ns1:POOL_KEYS">
                  <ns1:name>childPool-A</ns1:name>
                  <ns1:parentName>seedPool-A</ns1:parentName>
               </ns1:pool>
               <ns1:addressType>IPv6</ns1:addressType>
               <ns1:networkContact>networkContact@mycompany.com
  </ns1:networkContact>
               <ns1:ruleName>v6UsedRule</ns1:ruleName>
               <ns1:rirReport>
                  <ns1:reportType>ARIN-IPv6-Reassign</ns1:reportType>
                  <ns1:arinReport>
                     <ns1:orgInfo>
                        <ns1:fullName>orgName</ns1:fullName>
                        <ns1:address>600 Mountain Avenue</ns1:address>
                        <ns1:city>Murray Hill</ns1:city>
                        <ns1:stateOrProvince>NJ</ns1:stateOrProvince>
                        <ns1:postalCode>07974</ns1:postalCode>
                        <ns1:countryCode>US</ns1:countryCode>
                     </ns1:orgInfo>
                     <ns1:orgContactInfo>
                        <ns1:contactType>P</ns1:contactType>
                        <ns1:lastNameOrRoleAccount>my org POC last name
        </ns1:lastNameOrRoleAccount>
                        <ns1:firstName>my org POC First Name
</ns1:firstName>
```

```
<ns1:companyName>my company name </ns1:companyName>
                        <ns1:address>600 Mountain Avenue</ns1:address>
                        <ns1:city>Murray Hill</ns1:city>
                        <ns1:stateOrProvince>NJ</ns1:stateOrProvince>
                        <ns1:postalCode>07974</ns1:postalCode>
                        <ns1:countryCode>US</ns1:countryCode>
                        <ns1:phoneNumber>891-345-5678</ns1:phoneNumber>

   <ns1:emailAddress>orgPOCEmail@mycompany.com</ns1:emailAddress>

   <ns1:emailAddress2>alternateOrgPOCEmail@mycompany.com</ns1:emailAddress2>
                     </ns1:orgContactInfo>
                     <ns1:networkContactInfo>
                        <ns1:contactType>P</ns1:contactType>
                        <ns1:lastNameOrRoleAccount>net POC Last Name
</ns1:lastNameOrRoleAccount>
                        <ns1:firstName>net contact first name
</ns1:firstName>
                        <ns1:companyName>my company name</ns1:companyName>
                        <ns1:address>600 Mountain Avenue</ns1:add3d5ress>
                        <ns1:city>Murray Hill</ns1:city>
                        <ns1:stateOrProvince>NJ</ns1:stateOrProvince>
                        <ns1:postalCode>07974</ns1:postalCode>
                        <ns1:countryCode>US</ns1:countryCode>
                        <ns1:phoneNumber>891-234-4567</ns1:phoneNumber>

   <ns1:emailAddress>netContactEmail@mycompany.com</ns1:emailAddress>

   <ns1:emailAddress2>netContactEmail2@mycompany.com</ns1:emailAddress2>
                     </ns1:networkContactInfo>
                     <ns1:originAs>192.168.0.0</ns1:originAs>

   <ns1:dnsReverseMappingNameserver1>dnsServer1.mycompany.com</ns1:dnsRevers
   eMappingNameserver1>

   <ns1:dnsReverseMappingNameserver2>dnsServer2.mycompany.com</ns1:dnsRevers
   eMappingNameserver2>
                    <ns1:justification>justification text</ns1:justification>
                      <ns1:netPOCType>T</ns1:netPOCType>
                     <ns1:publicComments>This is a sample block allocated via
   VitalQIP API.</ns1:publicComments>
                      <ns1:additionalInformation>this is additional
   information</ns1:additionalInformation>
                  </ns1:arinReport>
               </ns1:rirReport>
            </ns1:parameters>
         </ns1:AllocateBlockToPoolMessageRequest>
```

```
      </soapenv:Body>
   </soapenv:Envelope>
```

Response

```
<?xml version='1.0' encoding='UTF-8'?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Body>
         <ns1:BlockAllocResponse xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:BlockAllocResponse">
            <ns1:result>SUCCESS</ns1:result>
            <ns1:pool xsi:type="ns1:POOL_KEYS">
               <ns1:name>childPool-A</ns1:name>
               <ns1:parentName>seedPool-A</ns1:parentName>
            </ns1:pool>
            <ns1:block xsi:type="ns1:ADDRESS_BLOCK_KEYS">
               <ns1:startAddress>2:3::1:0:0</ns1:startAddress>
               <ns1:prefixLength>96</ns1:prefixLength>
               <ns1:addressType>IPv6</ns1:addressType>
            </ns1:block>
         </ns1:BlockAllocResponse>
      </soapenv:Body>
   </soapenv:Envelope>
```

VQIPManager_AssociateRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Header>
```

… Refer to "Incoming/outgoing message structure format" (p. 2-6) for details.

```
      </soapenv:Header>
      <soapenv:Body>
         <ns1:AssociateRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:AssociateRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:srcKey xsi:type="ns1:RULE_KEYS">
               <ns1:name>v4UsedRule27</ns1:name>
            </ns1:srcKey>
            <ns1:targetKey xsi:type="ns1:SUBNET_PROF_TEMPL_KEYS">
               <ns1:name>QIP_SUBNET_PROFILE_TMPL</ns1:name>
            </ns1:targetKey>
         </ns1:AssociateRequest>
      </soapenv:Body>
   </soapenv:Envelope>
```

Response

```
<?xml version='1.0' encoding='UTF-8'?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Body>
         <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
            <ns1:result>SUCCESS</ns1:result>
            <ns1:responseData />
         </ns1:Response>
      </soapenv:Body>
   </soapenv:Envelope>
```

VQIPManager_AssociateListRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:wsa="http://www.w3.org/2005/08/addressing">
      <soapenv:Header>
        … Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
      </soapenv:Header>
      <soapenv:Body>
         <ns1:AssociateListRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:AssociateListRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:srcKeys xsi:type="ns1:UDA_DEFINITION_KEY">
               <ns1:name>uda1</ns1:name>
            </ns1:srcKeys>
            <ns1:srcKeys xsi:type="ns1:UDA_DEFINITION_KEY">
               <ns1:name>uda2</ns1:name>
            </ns1:srcKeys>
            <ns1:targetKey xsi:type="ns1:UDA_INFRASTRUCTURE_TYPE_KEY">
               <ns1:name>MANAGED_FILE</ns1:name>
            </ns1:targetKey>
         </ns1:AssociateListRequest>
      </soapenv:Body>
   </soapenv:Envelope>
```

Response

```
<?xml version='1.0' encoding='UTF-8'?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:wsa="http://www.w3.org/2005/08/addressing">
      <soapenv:Header>
         <wsa:Action>http://alcatel-
   lucent.com/qip/nb/ws/VQIPManagerPortType/VQIPManager_AssociateListRequest
   Response</wsa:Action>

   <wsa:RelatesTo>urn:uuid:2F09F5C38A1B3F81651248286437473</wsa:RelatesTo>
      </soapenv:Header>
      <soapenv:Body>
```

```
      <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="ns1:Response">
         <ns1:result>SUCCESS</ns1:result>
         <ns1:responseData />
      </ns1:Response>
   </soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_CancelRequest operation

Request

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:CancelRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:CancelRequest">
       <ns1:commonParam>
         <ns1:organization>QIP_MUSIC_COM</ns1:organization>
         <ns1:locale>en_US</ns1:locale>
       </ns1:commonParam>
       <ns1:reqObject xsi:type="ns1:SCHEDULE_JOB_REC">
         <ns1:id>51</ns1:id>
       </ns1:reqObject>
    </ns1:CancelRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="ns1:Response">
      <ns1:result>SUCCESS</ns1:result>
    </ns1:Response>
  </soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_CopyRequest operation

Request

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:CopyRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:CopyRequest">
      <ns1:commonParam>
        <ns1:organization>VitalQIP Organization</ns1:organization>
        <ns1:locale>en_US</ns1:locale>
      </ns1:commonParam>
      <ns1:from xsi:type="ns1:DNS_ACL_TEMPLATE_KEY">
        <ns1:name>acl1</ns1:name>
      </ns1:from>
      <ns1:to xsi:type="ns1:DNS_ACL_TEMPLATE_KEY">
        <ns1:name>acl2</ns1:name>
      </ns1:to>
    </ns1:CopyRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
      <ns1:result>SUCCESS</ns1:result>
      <ns1:responseData />
    </ns1:Response>
  </soapenv:Body>
</soapenv:Envelope>
```

Request for Clone DNS Server

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6).
```

```
</soapenv:Header>
      <soapenv:Body>
         <ns1:CopyRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance
   xsi:type="ns1:CopyRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:jobReq>
               <ns1:sendEmail>false</ns1:sendEmail>
               <ns1:runMode>IN_BACKGROUND</ns1:runMode>
            </ns1:jobReq>
            <ns1:from xsi:type="ns1:SERVER_KEY">
               <ns1:host>qdev3vm72pf1</ns1:host>
               <ns1:domain>qlab.lucent.com</ns1:domain>
               <ns1:serverType>DNS</ns1:serverType>
            </ns1:from>
               <ns1:to xsi:type="ns1:SERVER_KEY">
               <ns1:host>sharmila2</ns1:host>
               <ns1:domain>qlab.lucent.com</ns1:domain>
               <ns1:serverType>DNS</ns1:serverType>
            </ns1:to>
            <ns1:addlParams>
               <ns1:name>INCLUDEZONES</ns1:name>
            </ns1:addlParams>
            <ns1:addlParams>
               <ns1:name>MAKESECONDARY</ns1:name>
            </ns1:addlParams>
            </ns1:CopyRequest>
      </soapenv:Body>
   </soapenv:Envelope>
```

Response

```
<?xml version='1.0' encoding='utf-8'?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Body>
         <ns1:Response xmlns:ns1=http://alcatel-lucent.com/qip/nb/ws
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response"><ns1:result>SUCCESS</ns1:result>
         </ns1:Response>
      </soapenv:Body>
   </soapenv:Envelope>
```

VQIPManager_CopyRuleRequest operation

Request

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:CopyRuleRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:CopyRuleRequest">
       <ns1:commonParam>
         <ns1:organization>VitalQIP Organization</ns1:organization>
         <ns1:locale>en_US</ns1:locale>
       </ns1:commonParam>
       <ns1:fromRule xsi:type="ns1:RULE_REC">
         <ns1:name>QIP_V6_SIZE_40_USED</ns1:name>
       </ns1:fromRule>
       <ns1:toRule>NEW_QIP_V6_SIZE_40_USED</ns1:toRule>
    </ns1:CopyRuleRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
       <ns1:result>SUCCESS</ns1:result>
       <ns1:responseData />
    </ns1:Response>
  </soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_CountAssociationsRequest operation

Request

```
<?xml version='1.0' encoding='utf-8'?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
         <ns1:CountAssociationsRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:CountAssociationsRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:objectKey xsi:type="ns1:SUBNET_PROF_TEMPL_KEYS">
               <ns1:name>QIP_SUBNET_PROFILE_TMPL</ns1:name>
            </ns1:objectKey>
         </ns1:CountAssociationsRequest>
      </soapenv:Body>
   </soapenv:Envelope>
```

Response

```
<?xml version='1.0' encoding='UTF-8'?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Body>
         <ns1:CountAssociationsResponse xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:CountAssociationsResponse">
            <ns1:result>SUCCESS</ns1:result>
            <ns1:count>3</ns1:count>
         </ns1:CountAssociationsResponse>
      </soapenv:Body>
   </soapenv:Envelope>
```

VQIPManager_DeleteContainedObjectRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:DeleteContainedObjectRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:DeleteContainedObjectRequest">
      <ns1:commonParam>
        <ns1:organization>VitalQIP Organization</ns1:organization>
        <ns1:locale>en_US</ns1:locale>
      </ns1:commonParam>
      <ns1:objectKey xsi:type="ns1:INTERFACE_KEYS">
        <ns1:name>interface3</ns1:name>
        <ns1:nodeKey xsi:type="ns1:NODE_KEYS">
          <ns1:uniqueID>node1</ns1:uniqueID>
        </ns1:nodeKey>
      </ns1:objectKey>
      <ns1:deleteRelatedRecords>true</ns1:deleteRelatedRecords>
    </ns1:DeleteContainedObjectRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
      <ns1:result>SUCCESS</ns1:result>
      <ns1:responseData />
    </ns1:Response>
  </soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_DeleteForcedRequest operation

Request

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:DeleteForcedRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:DeleteForcedRequest">
       <ns1:commonParam>
         <ns1:organization>QIP_MUSIC_COM</ns1:organization>
         <ns1:locale>en_US</ns1:locale>
       </ns1:commonParam>
       <ns1:objectKey xsi:type="ns1:MANAGED_FILE_KEY">
         <ns1:name>myManagedFile</ns1:name>
       </ns1:objectKey>
       <ns1:forced>false</ns1:forced>
    </ns1:DeleteForcedRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version='1.0' encoding='UTF-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <soapenv:Fault>
      <faultcode>soapenv:Server</faultcode>
      <faultstring>ApplFaultException</faultstring>
      <detail>
        <ns1:ApplFault xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws">
          <ns1:errorMsg>Unable to delete Managed File - Managed File is in
   use</ns1:errorMsg>
          <ns1:errorKey>MANAGED_FILE_NO_DELETE_INUSE</ns1:errorKey>
        </ns1:ApplFault>
      </detail>
    </soapenv:Fault>
  </soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_DeleteSeedBlockFromSeedPoolRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:DeleteSeedBlockFromSeedPoolRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:DeleteSeedBlockFromSeedPoolRequest">
      <ns1:commonParam>
        <ns1:organization>VitalQIP Organization</ns1:organization>
        <ns1:locale>en_US</ns1:locale>
      </ns1:commonParam>
      <ns1:addressBlock xsi:type="ns1:ADDRESS_BLOCK_REC">
        <ns1:startAddress>192.168.1.0</ns1:startAddress>
        <ns1:prefixLength>24</ns1:prefixLength>
        <ns1:pool xsi:type="ns1:POOL_KEYS">
          <ns1:name>seedPool-A</ns1:name>
        </ns1:pool>
      </ns1:addressBlock>
    </ns1:DeleteSeedBlockFromSeedPoolRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
      <ns1:result>SUCCESS</ns1:result>
      <ns1:responseData />
    </ns1:Response>
  </soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_DissociateRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
         <ns1:DissociateRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:DissociateRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:srcKey xsi:type="ns1:RULE_KEYS">
               <ns1:name>v4UsedRule27</ns1:name>
            </ns1:srcKey>
            <ns1:targetKey xsi:type="ns1:SUBNET_PROF_TEMPL_KEYS">
               <ns1:name>QIP_SUBNET_PROFILE_TMPL</ns1:name>
            </ns1:targetKey>
         </ns1:DissociateRequest>
      </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Body>
         <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
            <ns1:result>SUCCESS</ns1:result>
            <ns1:responseData />
         </ns1:Response>
      </soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_DissociateListRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:wsa="http://www.w3.org/2005/08/addressing">
      <soapenv:Header>
        …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
      </soapenv:Header>
      <soapenv:Body>
         <ns1:DissociateListRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:DissociateListRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:srcKey xsi:type="ns1:UDA_DEFINITION_KEY">
               <ns1:name>uda1</ns1:name>
            </ns1:srcKey>
            <ns1:targetKey xsi:type="ns1:UDA_INFRASTRUCTURE_TYPE_KEY">
               <ns1:name>MANAGED_FILE</ns1:name>
            </ns1:targetKey>
         </ns1:DissociateListRequest>
      </soapenv:Body>
   </soapenv:Envelope>
```

Response

```
<?xml version='1.0' encoding='UTF-8'?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:wsa="http://www.w3.org/2005/08/addressing">
      <soapenv:Header>
         <wsa:Action>http://alcatel-
   lucent.com/qip/nb/ws/VQIPManagerPortType/VQIPManager_DissociateListReques
   tResponse</wsa:Action>

   <wsa:RelatesTo>urn:uuid:2F09F5C38A1B3F81651248286697060</wsa:RelatesTo>
      </soapenv:Header>
      <soapenv:Body>
         <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
```

```
            <ns1:result>SUCCESS</ns1:result>
            <ns1:responseData />
         </ns1:Response>
      </soapenv:Body>
   </soapenv:Envelope>
```

VQIPManager_ExpandBlockRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
         <ns1:ExpandBlockRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:ExpandBlockRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:addressBlock xsi:type="ns1:ADDRESS_BLOCK_REC">
               <ns1:startAddress>192.168.0.0</ns1:startAddress>
               <ns1:prefixLength>28</ns1:prefixLength>
               <ns1:pool xsi:type="ns1:POOL_KEYS">
                  <ns1:name>childPool-B</ns1:name>
                  <ns1:parentName>seedPool-A</ns1:parentName>
               </ns1:pool>
            </ns1:addressBlock>
         </ns1:ExpandBlockRequest>
      </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Body>
         <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
            <ns1:result>SUCCESS</ns1:result>
            <ns1:responseData />
         </ns1:Response>
      </soapenv:Body>
   </soapenv:Envelope>
```

VQIPManager_ExplicitBlockAllocRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
         <ns1:ExplicitBlockAllocMessageRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:ExplicitBlockAllocMessageRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:parameters>
               <ns1:startAddress>192.168.3.96</ns1:startAddress>
               <ns1:prefixLength>28</ns1:prefixLength>
               <ns1:pool xsi:type="ns1:POOL_KEYS">
                  <ns1:name>childPool-B</ns1:name>
                  <ns1:parentName>seedPool-B</ns1:parentName>
               </ns1:pool>
               <ns1:blockStatus>USED</ns1:blockStatus>
               <ns1:addressType>IPv4</ns1:addressType>

   <ns1:subnetCreationOption>CREATE_NEW_IF_NECESSARY</ns1:subnetCreationOpti
   on>

   <ns1:networkContact>networkContact@mycompany.com</ns1:networkContact>
               <ns1:allocationRuleName>v6UsedRule</ns1:allocationRuleName>
               <ns1:useDefaultRuleSize>false</ns1:useDefaultRuleSize>
               <ns1:rirReport>
                  <ns1:reportType>INETNUM for IPv4</ns1:reportType>
                  <ns1:ripeReport>

   <ns1:networkNameOption>USER_DEFINED</ns1:networkNameOption>
                     <ns1:descr>report description</ns1:descr>
                     <ns1:country>FR</ns1:country>
                     <ns1:status>ALLOCATED PA</ns1:status>
                     <ns1:adminC>AUTO-2</ns1:adminC>
                     <ns1:techC>AUTO-3</ns1:techC>
                     <ns1:org>AUTO-1</ns1:org>
                     <ns1:netName>my net name 1</ns1:netName>
```

```
                    <ns1:revSrv>revSrv1.mycompany.com</ns1:revSrv>
                    <ns1:remarks>general remarks</ns1:remarks>
                    <ns1:mntByPassword>qipman</ns1:mntByPassword>
                    <ns1:mntLower>mnt-lower</ns1:mntLower>
                    <ns1:mntLowerPassword>mntLower
password</ns1:mntLowerPassword>
                    <ns1:mntDomains>mntDomain123</ns1:mntDomains>
                    <ns1:mntRoutes>mntRouter123</ns1:mntRoutes>
                    <ns1:mntIrt>mntIrt123</ns1:mntIrt>
                    <ns1:notify>admin@mycompany.com</ns1:notify>
                    <ns1:changed>user1@mycompany.com</ns1:changed>
                    <ns1:source>RIPE</ns1:source>
                    <ns1:organisationObject>
                       <ns1:orgName>VitalQIP Organization</ns1:orgName>
                       <ns1:orgType>OTHER</ns1:orgType>
                       <ns1:address>54 rue de la Bo?6e9??tie</ns1:address>
                       <ns1:email>orgEmail@mycompany.com</ns1:email>
                       <ns1:mntRef>mnt-ref</ns1:mntRef>
                     <ns1:descr>organization section description</ns1:descr>
                       <ns1:remarks>organization remarks</ns1:remarks>
                       <ns1:phone>851-222-3333</ns1:phone>
                       <ns1:faxNo>851-222-3334</ns1:faxNo>
                       <ns1:refNfy>refNfy</ns1:refNfy>
                       <ns1:notify>orgNotify@mycompany.com</ns1:notify>

<ns1:abuseMailbox>orgAbuseMailbox@mycompany.com</ns1:abuseMailbox>

<ns1:changed>orgChangedEmail@mycompany.com</ns1:changed>
                       <ns1:source>RIPE</ns1:source>
                    </ns1:organisationObject>
                    <ns1:adminContactObject>
                       <ns1:person>Joe Clark</ns1:person>
                       <ns1:address>54 rue de la Bo?tie</ns1:address>
                       <ns1:phone>33 1 40 76 10 10</ns1:phone>
                       <ns1:country>FR</ns1:country>
                       <ns1:faxNo>33 1 40 76 10 11</ns1:faxNo>
                       <ns1:email>adminEmail@mycompany.com</ns1:email>
                       <ns1:remarks>admin section remarks</ns1:remarks>
                       <ns1:notify>adminNotify@mycompany.com</ns1:notify>

<ns1:abuseMailbox>adminAbuseMailbox@mycompany.com</ns1:abuseMailbox>
                       <ns1:mntBy>mntBy123</ns1:mntBy>
                      <ns1:changed>adminChanged@mycompany.com</ns1:changed>
                       <ns1:source>RIPE</ns1:source>
                    </ns1:adminContactObject>
                    <ns1:techContactObject>
```

```
                        <ns1:person>Techie Smith</ns1:person>
                        <ns1:address>54 rue de la Boetie</ns1:address>
                        <ns1:phone>33 1 40 76 10 11</ns1:phone>
                        <ns1:country>FR</ns1:country>
                        <ns1:faxNo>33 1 40 76 10 12</ns1:faxNo>
                        <ns1:email>techEmail@mycompany.com</ns1:email>
                        <ns1:remarks>tech section remarks</ns1:remarks>
                        <ns1:notify>techNotify@mycompany.com</ns1:notify>

   <ns1:abuseMailbox>techAbuseMailbox@mycompany.com</ns1:abuseMailbox>
                        <ns1:mntBy>techMntBy</ns1:mntBy>
                        <ns1:changed>techChanged@mycompany.com</ns1:changed>
                        <ns1:source>RIPE</ns1:source>
                     </ns1:techContactObject>
                  </ns1:ripeReport>
               </ns1:rirReport>
            </ns1:parameters>
         </ns1:ExplicitBlockAllocMessageRequest>
      </soapenv:Body>
   </soapenv:Envelope
```

Response

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Body>
 <ns1:BlockAllocResponse xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:type="ns1:BlockAllocResponse">
    <ns1:result>SUCCESS</ns1:result>
    <ns1:pool xsi:type="ns1:POOL_KEYS">
       <ns1:name>childPool-B</ns1:name>
       <ns1:parentName>seedPool-B</ns1:parentName>
    </ns1:pool>
    <ns1:block xsi:type="ns1:ADDRESS_BLOCK_KEYS">
       <ns1:startAddress>192.168.3.96</ns1:startAddress>
       <ns1:prefixLength>28</ns1:prefixLength>
       <ns1:addressType>IPv4</ns1:addressType>
    </ns1:block>
 </ns1:BlockAllocResponse>
</soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_FindAssociationsRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:FindAssociationsRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:FindAssociationsRequest">
       <ns1:commonParam>
         <ns1:organization>QIP_MUSIC_COM</ns1:organization>
         <ns1:locale>en_US</ns1:locale>
       </ns1:commonParam>
       <ns1:objectKey xsi:type="ns1:MANAGED_FILE_KEY">
         <ns1:name>myManagedFile</ns1:name>
       </ns1:objectKey>
    </ns1:FindAssociationsRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:FindAssociationsResponse xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:FindAssociationsResponse">
       <ns1:result>SUCCESS</ns1:result>
       <ns1:objects xsi:type="ns1:MANAGED_FILE_SERVER_KEY">
         <ns1:name>dnsmusic1.domain1.music.com</ns1:name>
         <ns1:type>DNS</ns1:type>
       </ns1:objects>
       <ns1:objects xsi:type="ns1:MANAGED_FILE_SERVER_KEY">
         <ns1:name>dnsmusic2.domain2.music.com</ns1:name>
         <ns1:type>DNS</ns1:type>
       </ns1:objects>
    </ns1:FindAssociationsResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_FreeBlockRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
         <ns1:FreeBlockRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:FreeBlockRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:block xsi:type="ns1:ADDRESS_BLOCK_REC">
               <ns1:startAddress>192.168.0.16</ns1:startAddress>
               <ns1:prefixLength>28</ns1:prefixLength>
               <ns1:pool xsi:type="ns1:POOL_KEYS">
                  <ns1:name>childPool-B</ns1:name>
                  <ns1:parentName>seedPool-A</ns1:parentName>
               </ns1:pool>
            </ns1:block>
            <ns1:returnToParent>false</ns1:returnToParent>
         </ns1:FreeBlockRequest>
      </soapenv:Body>
   </soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="utf-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Body>
         <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
            <ns1:result>SUCCESS</ns1:result>
            <ns1:responseData />
         </ns1:Response>
      </soapenv:Body>
   </soapenv:Envelope>
```

VQIPManager_FreePendingBlockRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
         <ns1:FreePendingBlockRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:FreePendingBlockRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:level>ALL</ns1:level>
            <ns1:override>false</ns1:override>
         </ns1:FreePendingBlockRequest>
      </soapenv:Body>
   </soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="utf-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Body>
         <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
            <ns1:result>SUCCESS</ns1:result>
            <ns1:responseData />
         </ns1:Response>
      </soapenv:Body>
   </soapenv:Envelope>
```

VQIPManager_GetAllRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:GetAllRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:GetAllRequest">
       <ns1:commonParam>
         <ns1:organization>VitalQIP Organization</ns1:organization>
         <ns1:locale>en_US</ns1:locale>
       </ns1:commonParam>
       <ns1:listType>OBJ_CLASS</ns1:listType>
    </ns1:GetAllRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:GetAllResponse xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:GetAllResponse">
       <ns1:result>SUCCESS</ns1:result>
       <ns1:list xsi:type="ns1:LIST_REC">
         <ns1:names>Workstation</ns1:names>
         <ns1:names>X-terminal</ns1:names>
         <ns1:names>PC</ns1:names>
         <ns1:names>Printer</ns1:names>
         <ns1:names>Server</ns1:names>
         <ns1:names>Wiring_HUB</ns1:names>
         <ns1:names>Router</ns1:names>
         <ns1:names>Bridge</ns1:names>
         <ns1:names>Terminal_Server</ns1:names>
         <ns1:names>Switch</ns1:names>
         <ns1:names>Legacy_System</ns1:names>
         <ns1:names>Gateway</ns1:names>
         <ns1:names>Test_Equipment</ns1:names>
         <ns1:names>Undefined</ns1:names>
```

```
        <ns1:names>Others</ns1:names>
        <ns1:names>Partially_Managed</ns1:names>
      </ns1:list>
    </ns1:GetAllResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_ GetAssociatedComponentsRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:wsa="http://www.w3.org/2005/08/addressing">
      <soapenv:Header>
         …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
      </soapenv:Header>
      <soapenv:Body>
         <ns1:GetAssociatedComponentsRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:GetAssociatedComponentsRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:param xsi:type="ns1:UDA_INFRASTRUCTURE_TYPE_KEY">
               <ns1:name>MANAGED_FILE</ns1:name>
            </ns1:param>
         </ns1:GetAssociatedComponentsRequest>
      </soapenv:Body>
   </soapenv:Envelope>
```

Response

```
<?xml version='1.0' encoding='UTF-8'?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:wsa="http://www.w3.org/2005/08/addressing">
      <soapenv:Header>
         <wsa:Action>http://alcatel-
   lucent.com/qip/nb/ws/VQIPManagerPortType/VQIPManager_GetAssociatedCompone
   ntsRequestResponse</wsa:Action>

   <wsa:RelatesTo>urn:uuid:2F09F5C38A1B3F81651248286557113</wsa:RelatesTo>
      </soapenv:Header>
      <soapenv:Body>
         <ns1:GetAssociatedComponentsResponse xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:GetAssociatedComponentsResponse">
            <ns1:result>SUCCESS</ns1:result>
            <ns1:items xsi:type="ns1:UDA_DEFINITION_LIST_REC">
               <ns1:uda>uda1</ns1:uda>
               <ns1:uda>uda2</ns1:uda>
```

```
            </ns1:items>
         </ns1:GetAssociatedComponentsResponse>
      </soapenv:Body>
   </soapenv:Envelope>
```

VQIPManager_GetContainedObjectsRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:GetContainedObjectsRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:GetContainedObjectsRequest">
      <ns1:commonParam>
        <ns1:organization>VitalQIP Organization</ns1:organization>
        <ns1:locale>en_US</ns1:locale>
      </ns1:commonParam>
      <ns1:parent xsi:type="ns1:POOL_KEYS">
        <ns1:name>seedPool-A</ns1:name>
      </ns1:parent>
      <ns1:containedObjectType>CHILD_POOL</ns1:containedObjectType>
    </ns1:GetContainedObjectsRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:GetContainedObjectsResponse xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:GetContainedObjectsResponse">
      <ns1:result>SUCCESS</ns1:result>
      <ns1:containedObjects xsi:type="ns1:POOL_KEYS">
        <ns1:name>childPool-B</ns1:name>
        <ns1:parentName>seedPool-A</ns1:parentName>
      </ns1:containedObjects>
      <ns1:containedObjects xsi:type="ns1:POOL_KEYS">
        <ns1:name>childPool-C</ns1:name>
        <ns1:parentName>seedPool-A</ns1:parentName>
      </ns1:containedObjects>
    </ns1:GetContainedObjectsResponse>
  </soapenv:Body>
</soapenv:Envelope
```

VQIPManager_GetFileRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:GetFileRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:GetFileRequest">
       <ns1:commonParam>
         <ns1:organization>VitalQIP Organization</ns1:organization>
         <ns1:locale>en_US</ns1:locale>
       </ns1:commonParam>
       <ns1:reqObject xsi:type="ns1:MANAGED_FILE_KEY">
         <ns1:name>myFile1</ns1:name>
       </ns1:reqObject>
       <ns1:os>UNIX</ns1:os>
    </ns1:GetFileRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:GetResponse xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:GetResponse">
       <ns1:result>SUCCESS</ns1:result>
       <ns1:record xsi:type="ns1:MANAGED_FILE_REC">
         <ns1:name>myFile1</ns1:name>
         <ns1:binaryData xmlns:ns2="http://www.w3.org/2005/05/xmlmime"
   ns2:contentType="application/octet-
   stream">2h1MZXhY3V0YWJsZSAoQ09NKQo=</ns1:binaryData>
       </ns1:record>
    </ns1:GetResponse>
  </soapenv:Body>
</soapenv:Envelope><?xml version='1.0' encoding='utf-8'?>
```

VQIPManager_LinkAddressToInterfaceRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:LinkAddressToInterfaceRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:LinkAddressToInterfaceRequest">
      <ns1:commonParam>
        <ns1:organization>QIP_MUSIC_COM</ns1:organization>
        <ns1:locale>en_US</ns1:locale>
      </ns1:commonParam>
      <ns1:ipAddrStr>21.0.0.9</ns1:ipAddrStr>
      <ns1:interfaceKey xsi:type="ns1:INTERFACE_KEYS">
        <ns1:name>eth0</ns1:name>
        <ns1:nodeKey xsi:type="ns1:NODE_KEYS">
          <ns1:uniqueID>252</ns1:uniqueID>
        </ns1:nodeKey>
      </ns1:interfaceKey>
    </ns1:LinkAddressToInterfaceRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
      <ns1:result>SUCCESS</ns1:result>
      <ns1:responseData />
    </ns1:Response>
  </soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_MergeRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
         <ns1:MergeRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:MergeRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:from xsi:type="ns1:ADDRESS_BLOCK_REC">
               <ns1:startAddress>192.168.0.0</ns1:startAddress>
               <ns1:prefixLength>28</ns1:prefixLength>
               <ns1:pool xsi:type="ns1:POOL_KEYS">
                  <ns1:name>childPool-B</ns1:name>
                  <ns1:parentName>seedPool-A</ns1:parentName>
               </ns1:pool>
            </ns1:from>
            <ns1:to xsi:type="ns1:ADDRESS_BLOCK_REC">
               <ns1:startAddress>192.168.0.16</ns1:startAddress>
               <ns1:prefixLength>28</ns1:prefixLength>
               <ns1:pool xsi:type="ns1:POOL_KEYS">
                  <ns1:name>childPool-B</ns1:name>
                  <ns1:parentName>seedPool-A</ns1:parentName>
               </ns1:pool>
            </ns1:to>
         </ns1:MergeRequest>
      </soapenv:Body>
   </soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="utf-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Body>
```

```
      <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="ns1:Response">
         <ns1:result>SUCCESS</ns1:result>
         <ns1:responseData />
      </ns1:Response>
   </soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_MoveObjectRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
         <ns1:MoveObjectRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:MoveObjectRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:objKey xsi:type="ns1:INTERFACE_KEYS">
               <ns1:name>Ethernet0</ns1:name>
               <ns1:nodeKey xsi:type="ns1:NODE_KEYS">
                  <ns1:uniqueID>node1</ns1:uniqueID>
               </ns1:nodeKey>
            </ns1:objKey>
            <ns1:toKey xsi:type="ns1:NODE_KEYS">
               <ns1:uniqueID>node2</ns1:uniqueID>
            </ns1:toKey>
         </ns1:MoveObjectRequest>
      </soapenv:Body>
   </soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Body>
         <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
            <ns1:result>SUCCESS</ns1:result>
            <ns1:responseData />
         </ns1:Response>
      </soapenv:Body>
   </soapenv:Envelope>
```

VQIPManager_MoveRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
         <ns1:MoveRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:MoveRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:source xsi:type="ns1:ADDRESS_BLOCK_REC">
               <ns1:startAddress>192.168.0.16</ns1:startAddress>
               <ns1:prefixLength>28</ns1:prefixLength>
               <ns1:pool xsi:type="ns1:POOL_KEYS">
                  <ns1:name>childPool-B</ns1:name>
                  <ns1:parentName>seedPool-A</ns1:parentName>
               </ns1:pool>
            </ns1:source>
            <ns1:target xsi:type="ns1:POOL_REC">
               <ns1:name>childPool-C</ns1:name>
               <ns1:parentName>seedPool-A</ns1:parentName>
            </ns1:target>
         </ns1:MoveRequest>
      </soapenv:Body>
   </soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Body>
         <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
            <ns1:result>SUCCESS</ns1:result>
            <ns1:responseData />
         </ns1:Response>
```

```
    </soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_QuickBlockAllocRequest operation

Request

```
<?xml version="1.0" encoding="utf-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
         <ns1:QuickBlockAllocMessageRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:QuickBlockAllocMessageRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:parameters>
               <ns1:parentPool xsi:type="ns1:POOL_KEYS">
                  <ns1:name>childPool-C</ns1:name>
                  <ns1:parentName>seedPool-C</ns1:parentName>
               </ns1:parentPool>
               <ns1:childPoolName>childPool-C2</ns1:childPoolName>
               <ns1:addressType>IPv4</ns1:addressType>
               <ns1:ruleName>v4UsedRule</ns1:ruleName>
               <ns1:poolContact>childPoolC2@mycompany.com</ns1:poolContact>

   <ns1:networkContact>networkContact@mycompany.com</ns1:networkContact>
               <ns1:rirReport>
                  <ns1:reportType>INETNUM for IPv4</ns1:reportType>
                  <ns1:apnicReport>

   <ns1:networkNameOption>USER_DEFINED</ns1:networkNameOption>
                     <ns1:netName>my net name 1</ns1:netName>
                     <ns1:descr>report description</ns1:descr>
                     <ns1:country>FR</ns1:country>
                     <ns1:adminC>AUTO-2</ns1:adminC>
                     <ns1:techC>AUTO-3</ns1:techC>
                     <ns1:revSrv>revSrv1.mycompany.com</ns1:revSrv>
                     <ns1:status>ALLOCATED PORTABLE</ns1:status>
                     <ns1:remarks>general remarks</ns1:remarks>
                     <ns1:mntByPassword>qipman</ns1:mntByPassword>
                     <ns1:mntLower>mnt-lower</ns1:mntLower>
                     <ns1:mntLowerPassword>mntLower
   password</ns1:mntLowerPassword>
```

```
                        <ns1:mntRoutes>mntRouter123</ns1:mntRoutes>
                        <ns1:notify>admin@mycompany.com</ns1:notify>
                        <ns1:changed>user1@mycompany.com</ns1:changed>
                        <ns1:source>APNIC</ns1:source>
                        <ns1:adminContactObject>
                           <ns1:person>Joe Clark</ns1:person>
                           <ns1:address>54 rue de la Bo?tie</ns1:address>
                           <ns1:phone>33 1 40 76 10 10</ns1:phone>
                           <ns1:country>FR</ns1:country>
                           <ns1:faxNo>33 1 40 76 10 11</ns1:faxNo>
                           <ns1:email>adminEmail@mycompany.com</ns1:email>
                           <ns1:remarks>admin section remarks</ns1:remarks>
                           <ns1:notify>adminNotify@mycompany.com</ns1:notify>
                           <ns1:mntBy>mntBy123</ns1:mntBy>2cf
                          <ns1:changed>adminChanged@mycompany.com</ns1:changed>
                           <ns1:source>APNIC</ns1:source>
                        </ns1:adminContactObject>
                        <ns1:techContactObject>
                           <ns1:person>Joe Smith</ns1:person>
                           <ns1:address>54 rue de la Boetie</ns1:address>
                           <ns1:phone>33 1 40 76 10 11</ns1:phone>
                           <ns1:country>FR</ns1:country>
                           <ns1:faxNo>33 1 40 76 10 12</ns1:faxNo>
                           <ns1:email>techEmail@mycompany.com</ns1:email>
                           <ns1:remarks>tech section remarks</ns1:remarks>
                           <ns1:notify>techNotify@mycompany.com</ns1:notify>
                           <ns1:mntBy>techMntBy</ns1:mntBy>
                           <ns1:changed>techChanged@mycompany.com</ns1:changed>
                           <ns1:source>APNIC</ns1:source>
                        </ns1:techContactObject>
                     </ns1:apnicReport>
                  </ns1:rirReport>
               </ns1:parameters>
            </ns1:QuickBlockAllocMessageRequest>
         </soapenv:Body>
      </soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Body>
         <ns1:BlockAllocResponse xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:BlockAllocResponse">
```

```
            <ns1:result>SUCCESS</ns1:result>
            <ns1:pool xsi:type="ns1:POOL_KEYS">
               <ns1:name>childPool-C2</ns1:name>
               <ns1:parentName>childPool-C</ns1:parentName>
            </ns1:pool>
            <ns1:block xsi:type="ns1:ADDRESS_BLOCK_KEYS">
               <ns1:startAddress>192.168.4.32</ns1:startAddress>
               <ns1:prefixLength>28</ns1:prefixLength>
               <ns1:addressType>IPv4</ns1:addressType>
            </ns1:block>
         </ns1:BlockAllocResponse>
      </soapenv:Body>
   </soapenv:Envelope>
```

VQIPManager_RecoverBlockRequest operation

Request

```
<?xml version="1.0" encoding="utf-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Header>
… Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
      </soapenv:Header>
      <soapenv:Body>
         <ns1:RecoverBlockMessageRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:RecoverBlockMessageRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:parameters>
               <ns1:block xsi:type="ns1:ADDRESS_BLOCK_KEYS">
                  <ns1:startAddress>192.168.0.0</ns1:startAddress>
                  <ns1:prefixLength>28</ns1:prefixLength>
                  <ns1:addressType>IPv4</ns1:addressType>
               </ns1:block>
               <ns1:rirReport>
                  <ns1:reportType>ARIN-Reassign-Simple</ns1:reportType>
                  <ns1:arinReassignSimpleReport>
                     <ns1:registrationAction>N</ns1:registrationAction>
                     <ns1:customerInfo>
                        <ns1:fullName>Alcatel-Lucent</ns1:fullName>
                        <ns1:address>600 Mountain Ave</ns1:address>
                        <ns1:city>Murray Hill</ns1:city>
                        <ns1:stateOrProvince>NJ</ns1:stateOrProvince>
                        <ns1:postalCode>07974</ns1:postalCode>
                        <ns1:countryCode>US</ns1:countryCode>
                     </ns1:customerInfo>
                     <ns1:usePrivateName>false</ns1:usePrivateName>
                     <ns1:usePrivateAddress>false</ns1:usePrivateAddress>
                      <ns1:publicComments>Recover pending block and send simple
   report to ARIN</ns1:publicComments>
                  </ns1:arinReassignSimpleReport>
               </ns1:rirReport>
            </ns1:parameters>
         </ns1:RecoverBlockMessageRequest>
      </soapenv:Body>
```

```
</soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Body>
         <ns1:RecoverBlockResponse xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:RecoverBlockResponse">
            <ns1:result>SUCCESS</ns1:result>
            <ns1:blockKey xsi:type="ns1:ADDRESS_BLOCK_KEYS">
               <ns1:startAddress>192.168.0.0</ns1:startAddress>
               <ns1:prefixLength>28</ns1:prefixLength>
            </ns1:blockKey>
         </ns1:RecoverBlockResponse>
      </soapenv:Body>
   </soapenv:Envelope>
```

VQIPManager_RenumberRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
        <ns1:RenumberRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:RenumberRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:jobReq>
               <ns1:sendEmail>false</ns1:sendEmail>
               <ns1:runMode>IN_FOREGROUND</ns1:runMode>
            </ns1:jobReq>
            <ns1:source xsi:type="ns1:V6_SUBNET_REC">
               <ns1:startAddress>2:3:0:2::</ns1:startAddress>
            </ns1:source>
            <ns1:target xsi:type="ns1:V6_SUBNET_REC">
               <ns1:startAddress>2:3:0:3::</ns1:startAddress>
            </ns1:target>
         </ns1:RenumberRequest>
      </soapenv:Body>
   </soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Body>
         <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
            <ns1:result>SUCCESS</ns1:result>
            <ns1:responseData>
               <ns1:jobId>8</ns1:jobId>
               <ns1:name>RENUMBER_V6_SUBNET</ns1:name>
               <ns1:completionTime>Fri Mar 28 10:36:01 EDT
   2008</ns1:completionTime>
```

```
            </ns1:responseData>
         </ns1:Response>
      </soapenv:Body>
   </soapenv:Envelope>
```

VQIPManager_ReturnToParentBlockRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
         <ns1:ReturnToParentBlockRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:ReturnToParentBlockRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:addressBlock xsi:type="ns1:ADDRESS_BLOCK_REC">
               <ns1:startAddress>192.168.0.0</ns1:startAddress>
               <ns1:prefixLength>28</ns1:prefixLength>
               <ns1:pool xsi:type="ns1:POOL_KEYS">
                  <ns1:name>childPool-B</ns1:name>
                  <ns1:parentName>seedPool-A</ns1:parentName>
               </ns1:pool>
            </ns1:addressBlock>
         </ns1:ReturnToParentBlockRequest>
      </soapenv:Body>
   </soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Body>
         <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
            <ns1:result>SUCCESS</ns1:result>
            <ns1:responseData />
         </ns1:Response>
      </soapenv:Body>
   </soapenv:Envelope>
```

VQIPManager_SequenceRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:SequenceRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:SequenceRequest">
       <ns1:commonParam>
         <ns1:organization>VitalQIP Organization</ns1:organization>
         <ns1:locale>en_US</ns1:locale>
       </ns1:commonParam>
       <ns1:reqObject xsi:type="ns1:DNS_VIEW_KEY">
         <ns1:name>view1</ns1:name>
       </ns1:reqObject>
       <ns1:reqObject xsi:type="ns1:DNS_VIEW_KEY">
         <ns1:name>view2</ns1:name>
       </ns1:reqObject>
    </ns1:SequenceRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
       <ns1:result>SUCCESS</ns1:result>
    </ns1:Response>
  </soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_SequenceAssociatedComponentsRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:wsa="http://www.w3.org/2005/08/addressing">
      <soapenv:Header>
        …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
      </soapenv:Header>
      <soapenv:Body>
         <ns1:SequenceAssociatedComponentsRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:SequenceAssociatedComponentsRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:key xsi:type="ns1:UDA_INFRASTRUCTURE_TYPE_KEY">
               <ns1:name>MANAGED_FILE</ns1:name>
            </ns1:key>
            <ns1:components xsi:type="ns1:UDA_DEFINITION_LIST_REC">
               <ns1:uda>uda2</ns1:uda>
               <ns1:uda>uda1</ns1:uda>
            </ns1:components>
         </ns1:SequenceAssociatedComponentsRequest>
      </soapenv:Body>
   </soapenv:Envelope>
```

Response

```
<?xml version='1.0' encoding='UTF-8'?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:wsa="http://www.w3.org/2005/08/addressing">
      <soapenv:Header>
         <wsa:Action>http://alcatel-
   lucent.com/qip/nb/ws/VQIPManagerPortType/VQIPManager_SequenceAssociatedCo
   mponentsRequestResponse</wsa:Action>

   <wsa:RelatesTo>urn:uuid:2F09F5C38A1B3F81651248286622545</wsa:RelatesTo>
      </soapenv:Header>
      <soapenv:Body>
```

```
      <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="ns1:Response">
         <ns1:result>SUCCESS</ns1:result>
      </ns1:Response>
   </soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_SetUdaRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:wsa="http://www.w3.org/2005/08/addressing">
      <soapenv:Header>
         …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
      </soapenv:Header>
      <soapenv:Body>
         <ns1:SetUdaRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:SetUdaRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:infrastructure xsi:type="ns1:MANAGED_FILE_KEY">
               <ns1:name>myManagedFile</ns1:name>
            </ns1:infrastructure>
            <ns1:udaList>
               <ns1:udas>
                  <ns1:name>uda2</ns1:name>
                  <ns1:value>newValue</ns1:value>
               </ns1:udas>
            </ns1:udaList>
         </ns1:SetUdaRequest>
      </soapenv:Body>
   </soapenv:Envelope>
```

Response

```
<?xml version='1.0' encoding='UTF-8'?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:wsa="http://www.w3.org/2005/08/addressing">
      <soapenv:Header>
         <wsa:Action>http://alcatel-
   lucent.com/qip/nb/ws/VQIPManagerPortType/VQIPManager_SetUdaRequestRespons
   e</wsa:Action>

   <wsa:RelatesTo>urn:uuid:2F09F5C38A1B3F81651248286933102</wsa:RelatesTo>
      </soapenv:Header>
      <soapenv:Body>
```

```
      <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="ns1:Response">
         <ns1:result>SUCCESS</ns1:result>
         <ns1:responseData />
      </ns1:Response>
   </soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_SplitRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
         <ns1:SplitRequest xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:SplitRequest">
            <ns1:commonParam>
               <ns1:organization>VitalQIP Organization</ns1:organization>
               <ns1:locale>en_US</ns1:locale>
            </ns1:commonParam>
            <ns1:entity xsi:type="ns1:ADDRESS_BLOCK_REC">
               <ns1:startAddress>192.168.0.0</ns1:startAddress>
               <ns1:prefixLength>28</ns1:prefixLength>
               <ns1:pool xsi:type="ns1:POOL_KEYS">
                  <ns1:name>childPool-B</ns1:name>
                  <ns1:parentName>seedPool-A</ns1:parentName>
               </ns1:pool>
            </ns1:entity>
            <ns1:requestedSize>29</ns1:requestedSize>
         </ns1:SplitRequest>
      </soapenv:Body>
   </soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
      <ns1:result>SUCCESS</ns1:result>
    </ns1:Response>
  </soapenv:Body>
</soapenv:Envelope>
```

VQIPManager_UNlinkAddressFromInterfaceRequest operation

Request

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Header>
    …Refer to "Incoming/outgoing message structure format" (p. 2-6) for
   details.
</soapenv:Header>
<soapenv:Body>
    <ns1:UnlinkAddressFromInterfaceRequest xmlns:ns1="http://alcatel-
   lucent.com/qip/nb/ws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xsi:type="ns1:UnlinkAddressFromInterfaceRequest">
       <ns1:commonParam>
         <ns1:organization>QIP_MUSIC_COM</ns1:organization>
         <ns1:locale>en_US</ns1:locale>
       </ns1:commonParam>
       <ns1:ipAddrStr>21.0.0.9</ns1:ipAddrStr>
       <ns1:interfaceKey xsi:type="ns1:INTERFACE_KEYS">
         <ns1:name>eth0</ns1:name>
         <ns1:nodeKey xsi:type="ns1:NODE_KEYS">
           <ns1:uniqueID>252</ns1:uniqueID>
         </ns1:nodeKey>
       </ns1:interfaceKey>
    </ns1:UnlinkAddressFromInterfaceRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:Response xmlns:ns1="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="ns1:Response">
       <ns1:result>SUCCESS</ns1:result>
       <ns1:responseData />
    </ns1:Response>
  </soapenv:Body>
</soapenv:Envelope>
```

5 Error messages

Overview

Purpose

This chapter lists the various error messages that may be received by the northbound system.

Contents

This chapter covers these topics.

Error messages

Overview

The VitalQIP Web Service populates the appropriate error messages in the **`ApplFault`** structure.

A list of possible error messages is given below:

- `REQUIRED_FLD_MISSING=Required Field(s) Missing: [{0}]`
- `WS_UNKNOWN_ERROR=Error returned by the service layer with no errorKey.`
- `WS_NOT_AVAILABLE=VitalQIP Service Layer is not reachable.`
- `WS_UNABLE_TO_GET_RULE=Rule not found in VitalQIP database.`
- `MSG_CTX_ERROR=Message context is null.`
- `WS_UNSUPPORTED_ENTITY=Request for unsupported entity: [{0}]`
- `ADDR_UNABLE_TO_COPY_RULE=Error during copying rule.`
- `WS_OPERATION_FAILED=Operation Failed: [{0}]`
- `WS_ADDRESS_BLOCK_NOT_FOUND=Address Block not found: [poolname={0}, parentname={1}, startAddr={2}, prefixLength={3}, status={4}]`
- `WS_ADDRESS_BLOCK_NOT_FOUND2=Address block {0}/{1} either does not exist or does not have USED status.`
- `WS_ADDRESS_BLOCK_NOT_PENDING=Address block {0}/{1} either does not exist or is not in pending state.`
- `WS_ADDRESS_RANGE_NOT_FOUND=Address Block not found: [name={0}, startAddr={1}, prefixLength={2}, parentBlockAddr={3}]`
- `WS_ADDRESS_BLOCK_MISSING_KEYS=Required Fields(s) Missing: either pool or block status must be specified.`
- `WS_ADDRESS_TEMPLATE_NOT_FOUND=Address Template not found: [name={0}]`
- `WS_DOMAIN_NOT_FOUND=Domain not found: [name={0}]`
- `WS_MAINTAINER_NOT_FOUND=Maintainer not found: [name={0}, registry={1}]`
- `WS_ORG_NOT_FOUND=Organization not found: [{0}]`
- `WS_POOL_NOT_FOUND=Pool not found: [name={0}, parent={1}]`
- `WS_REGISTRY_NOT_FOUND=Registry not found: [{0}]`
- `WS_REVERSEZONE_TEMPLATE_NOT_FOUND=Reverse Zone Template not found: [name={0}]`
- `WS_RULE_NOT_FOUND=Rule not found: [name={0}]`
- `WS_SUBNETORG_NOT_FOUND=Subnet Organization not found: [{0}]`
- `WS_SUBNET_PROFILE_TEMPLATE_NOT_FOUND=Subnet Profile Template not found: [name={0}]`

- WS_V6_SEED_BLOCK_NOT_FOUND=IPv6 Seed Block not found: [address={0}]
- WS_UDA_NOT_FOUND=User Defined Attribute not found: [name={0}].
- WS_UDAS_NOT_FOUND=User Defined Attributes not found: [names={0}].
- WS_UDAGROUP_NOT_FOUND=UDA group not found: [name={0}].
- WS_V6_BLOCK_NOT_SEED=Block [startAddr={0}] is not seed block.
- WS_POOL_NOT_SEED=Pool [name={0}, parent={1}] is not seed pool.
- WS_POOL_NOT_CHILD=Pool [name={0}, parent={1}] is not child pool.
- WS_INVALID_POOL_TYPE=Specified pool type is invalid for this operation.
- WS_POOL_NAME_NOT_UNIQUE=There is more than one address pool named {0}. Parent pool name is required.
- WS_SEARCH_CRITERIA=At least one search criteria must be specified.
- WS_INVALID_ACTION_TYPE=Invalid Action. Action [{0}] is not supported.
- WS_INVALID_ADDRESS_TYPE=Invalid IP Address Type. IP Address Type [{0}] is not supported.
- WS_INVALID_ALGORITHM=Invalid Allocation Algorithm. Algorithm [{0}] is not supported.
- WS_INVALID_BLOCK_ACTION_TYPE=Invalid Block Action. Action [{0}] is not supported.
- WS_INVALID_BLOCK_STATUS_TYPE=Invalid Address Block Status Type. Address Block Status Type [{0}] is not supported.
- WS_INVALID_BLOCK_STATUS_FOR_OPERATION=Specified Block Status is not valid for this operation [status={0}].
- WS_INVALID_BLOCK_TYPE=Invalid Address Block Type. Address Block Type [{0}] is not supported.
- WS_INVALID_BLOCK_FOR_DELETE=Unable to delete specified Address Block {0}/{1}. Address Block is not a seed block.
- WS_INVALID_ENCRYPTION_TYPE=Invalid Encryption Type. Encryption Type [{0}] is not supported for registry [{1}].
- WS_INVALID_INPUT=Invalid input specified for [{0}] parameter. Required [{1}], Received [{2}].
- WS_INVALID_OPERATION=Invalid operation. Requested operation is not valid for [{0}] entity.
- WS_INVALID_PENDING_CHECK_TYPE=Invalid Pending Block Check Type. Type [{0}] is not supported.
- WS_INVALID_REGISTRY_TYPE=Invalid Registry Type. Registry Type [{0}] is not supported.
- WS_NODE_NOT_FOUND=Node not found: [{0}]

- WS_INVALID_START_ADDRESS=Invalid start address: [{0}]
- WS_UNUSED_INPUT={0}
- WS_SUBNET_NOT_FOUND=Subnet not found: [{0}]
- WS_V6SUBNET_NOT_POPULATED=Subnet data not populated for record # [{0}]
- WS_UNKNOWN_HOST_EXCEPTION=Unknown host exception for address: [{0}]
- WS_SEED_BLOCK_NOT_FOUND=Subnet not found: [start address ={0}]
- WS_MISSING_SEARCH_CRITERIA=Required Field Missing: At least one search criterion should be populated.
- WS_INTERNAL_ERROR=Internal Error. Please contact System Administrator. [{0}]
- WS_INTERNAL_ERROR_CAUSE=cause: [{0}]
- WS_INTERFACE_NOT_FOUND=Interface not found: [{0}]
- WS_SITE_NOT_FOUND=Site not found: [{0}]
- WS_MISSING_VALUE=No value specified for: [{0}]
- WS_OBJECT_CLASS_NOT_FOUND=Object class not found for: [{0}]
- WS_RECORD_NOT_FOUND=Record not found: [{0}]
- WS_IPADDRESS_NOT_FOUND=IPAddress not found: [{0}]
- WS_INVALID_REPORT_TYPE=Invalid report type specified: [{0}]
- WS_MISSING_REPORT_TYPE=Report type not specified.
- WS_MISSING_REPORT_REC=Report record not specified.
- WS_REPORT_REQUIRED_FLD_MISSING=RIR Report not generated because required reports field missing: [{0}]
- WS_RIR_REPORT_FAILED=Generating RIR Reporting failed: [{0}]
- WS_CREATE_INFRASTRUCTURE_FAILED=Create infrastructure failed: [{0}]
- WS_DNS_SERVER_NOT_FOUND=DNS Server not found: [{0}]
- WS_TOO_MANY_ORG=Either Organization or OrgId should be specified, not both.
- WS_LOGIN_FAILED=Login Failed: Invalid UserName or Password.
- WS_LOGIN_MISSING=Username Token header missing.
- WS_DHCP_SERVER_NOT_FOUND=DHCP Server not found: [{0}]
- WS_SERVER_NOT_FOUND=Server not found: name=[{0}], type=[{1}]
- WS_SERVER_NOT_UNIQUE=Named server, [{0}], is not unique. Other matches: [{1}]
- WS_DHCP_OPT_TEMPL_NOT_FOUND=DHCP Option Template not found: [{0}]
- WS_INVALID_COUNTRY_CODE=Parameter [{0}] contains invalid country code [{1}].
- WS_INVALID_EMAIL_ADDRESS_FORMAT=Parameter [{0}] contains invalid email address [{1}].

- WS_INVALID_SEARCH_TYPE=Invalid node search type: [{0}]
- WS_IPV4_IPV6_SEARCH_NOT_SUPPORTED=Invalid search criteria. Search based on both IPv4 and IPv6 search criteria is not supported.
- WS_UDA_UDF_SEARCH_NOT_SUPPORTED=Invalid search criteria. Search based on both UDF and UDA criteria is not supported.
- WS_ONLY_ONE_ATTRIBUTE_ALLOWED=Invalid search criteria. Only one entry is allowed in ipAddressUdaList.
- WS_MISSING_ADDRESS_TYPE=Address Type must be specified for this request.
- WS_INVALID_BLOCK_SEARCH=Invalid block search. Start address or optional attributes can not be used when searching based on registry.
- WS_ADDR_ALREADY_UNLINKED=Address [{0}] is not linked to interface [{1}] in node with id [{2}].
- WS_BLOCK_EXPAND_FAILED=Specified Address Block either does not exist or it is not in an expandable state: [poolname={0}, parentname={1}, startAddr={2}, prefixLength={3}]
- WS_JOB_NOT_FOUND=Job not found: [id={0}]
- WS_FULL_NAME_IS_REQUIRED=Syntax error! Space between first and last name is required: [{0}]
- WS_ORG_MISSING_SITE=Organization {0} is not configured correctly. It is missing Site information.
- WS_IPV4_IPV6_REQUIRED=Could not determine whether requested IP address is IPv4 or IPv6.
- WS_DNS_ACL_TEMPLATE_NOT_FOUND=ACL not found: [name={0}]
- WS_MANAGED_FILE_NOT_FOUND=Managed File not found: [filename={0}]
- WS_IO_EXCEPTION=I/O Error: {0}
- WS_DNS_VIEW_NOT_FOUND=DNS View not found: [{0}]
- WS_DNS_ZONE_NOT_FOUND=DNS Zone not found: [{0}]
- WS_DNS_ZONE_IPV4REV_INVALID=An IPv4 reverse zone must also be a shared zone. [View={0} Zone={1}]
- WS_V4_OBJECT_NAME_NOT_FOUND=IPv4 Object''s name not found: [IP={0} FQDN={1} View={2}]
- WS_NOT_UPDATE_FIELD=An update request can not change the field: [{0}]
- WS_INVALID_COMBINATION=You can specify either {0} or {1}, but not both at the same time.
- WS_INVALID_COMBINATION_VALUE=[{0}] can not be specified if [{1}] is [{2}].
- WS_API_DISABLED=The web service VQIPManagerPort is disabled. Please contact System Administrator if this deprecated service needs to be enabled.

- WS_ALREADY_ASSOCIATED=[{0}] is already associated with [{1}].
- WS_INCONSISTENT_ADDRESS_TYPE=Inconsistent address type specified. If {0} is specified, it must match {1}.

Additionally, error messages initiated by VitalQIP Business Layer will be returned by the VitalQIP Web Service.

- WS_INVALID_UDA_BOOLEAN_DEFAULT=Invalid initialValue for boolean type. Valid values are true or false.
- WS_INVALID_UDA_TYPE=Invalid User Defined Attribute Type. Type [{0}] is not supported.
- WS_MULTIPLE_UDA_SEARCH_NOT_SUPPORTED=Invalid search criteria. Search using more than one UDA is not supported.
- WS_INVALID_SERVER_TYPE=Invalid server type: [{0}]
- WS_INVALID_ENTITY_FOR_OPERATION=Entity type [{0}] is not supported for this operation. Supported type is [{1}].
- WS_UDA_NOT_ENABLED=Infrastructure type {0} does not support UDAs.
- WS_MYVIEW_NOT_FOUND=MyView not found: [name={0}]
- WS_NETWORK_NOT_FOUND=Network not found: [address={0}]
- WS_DNS_ZONE_IPV4REV_NOT_FOUND=IPv4 Reverse Zone not found: [{0}]
- WS_VIEW_SPECIFIC_ZONE_NOT_FOUND=View Specific Zone not found: [View={0}, Zone={1}]
- WS_AC_DEVICE_NOT_FOUND=DHCP Device not found: [MAC Address={0}]
- WS_AC_SUBSCRIBER_NOT_FOUND=DHCP Subscriber not found: [Username={0}]
- WS_INVALID_AC_SUBSCRIBER_STATUS=Invalid DHCP Subscriber status. Status [{0}] is not supported.
- WS_ELEMENT_MUST_BE_NULL=The element [{0}] is not used here and must remain null.

6 VitalQIP Web Service schema

Overview

Purpose

This chapter describes the VitalQIP Web Service schema.

Contents

This chapter covers these topics.

Data types

Overview

The data types included below are used in the WSDL included in Chapter 7, "VitalQIP Web Service WSDL".

AcDevice.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Copyright (c) 2009 Alcatel-Lucent. -->
<xsd:schema
    xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
    xmlns:xi="http://www.w3.org/2001/XInclude"
    elementFormDefault="qualified">

    <xsd:include schemaLocation="CoreTypes.xsd" />

    <!-- define type AC_DEVICE_KEY -->
    <xsd:complexType name="AC_DEVICE_KEY">
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_KEY_ENTITY">
                <xsd:sequence>
                    <xsd:element name="macAddress"
  type="xsd:string"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
     </xsd:complexType>

     <!-- define type AC_DEVICE_REC -->
     <xsd:complexType name="AC_DEVICE_REC">
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_BASE_ENTITY">
                <xsd:sequence>
                    <xsd:element name="macAddress"
  type="xsd:string" minOccurs="0"/>
                    <xsd:element name="acSubscriber"
  type="xsd:string" minOccurs="0"/>
                  <xsd:element name="userClass" type="xsd:string"
  minOccurs="0"/>
                    <xsd:element name="optionalAttributeList"
  type="qipws:UdaList" minOccurs="0"/>

                    <!-- optional search related elements -->
                   <xsd:element name="hostname" type="xsd:string"
  minOccurs="0"/>
                    <xsd:element name="domain" type="xsd:string"
  minOccurs="0"/>
                   <xsd:element name="address" type="xsd:string"
  minOccurs="0"/>
```

```
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>

    <xsd:element name="AC_DEVICE_KEY" type="qipws:AC_DEVICE_KEY"/>
    <xsd:element name="AC_DEVICE_REC" type="qipws:AC_DEVICE_REC"/>

</xsd:schema>
```

AcSubscriber.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Copyright (c) 2009 Alcatel-Lucent. -->
<xsd:schema
    xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
    xmlns:xi="http://www.w3.org/2001/XInclude"
    elementFormDefault="qualified">

    <xsd:include schemaLocation="CoreTypes.xsd"/>

    <!-- define type AC_USERCLASS_LIST -->
    <xsd:complexType name="AC_USERCLASS_LIST">
        <xsd:sequence>
            <xsd:element name="userClass" type="xsd:string"
  minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
    </xsd:complexType>

   <!-- define type AC_USERCLASS_KEY, extends VQIP_BASE_ENTITY to
  be usable in Get request -->
    <xsd:complexType name="AC_USERCLASS_KEY">
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_BASE_ENTITY">
                <xsd:sequence>
                    <xsd:element name="acSubscriber"
  type="xsd:string"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>

    <!-- define type AC_USERCLASS_REC -->
    <xsd:complexType name="AC_USERCLASS_REC">
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_BASE_ENTITY">
                <xsd:sequence>
                    <xsd:element name="list"
  type="qipws:AC_USERCLASS_LIST" minOccurs="0"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>
```

```
 <!-- define enum AC_SUBSCRIBER_STATUS_TYPE -->
 <xsd:simpleType name="AC_SUBSCRIBER_STATUS_TYPE">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="ACTIVE"/>
  <xsd:enumeration value="INACTIVE"/>
  <xsd:enumeration value="PENDING"/>
 </xsd:restriction>
 </xsd:simpleType>

 <!-- define type AC_SUBSCRIBER_KEY -->
 <xsd:complexType name="AC_SUBSCRIBER_KEY">
     <xsd:complexContent>
         <xsd:extension base="qipws:VQIP_KEY_ENTITY">
             <xsd:sequence>
                 <xsd:element name="acSubscriber"
type="xsd:string"/>
             </xsd:sequence>
         </xsd:extension>
     </xsd:complexContent>
 </xsd:complexType>

 <!-- define type AC_SUBSCRIBER_REC -->
 <xsd:complexType name="AC_SUBSCRIBER_REC">
     <xsd:complexContent>
         <xsd:extension base="qipws:VQIP_BASE_ENTITY">
             <xsd:sequence>
                 <xsd:element name="acSubscriber"
type="xsd:string" minOccurs="0"/>
                 <xsd:element name="acSubscriberFirstName"
type="xsd:string" minOccurs="0"/>
                 <xsd:element name="acSubscriberLastName"
type="xsd:string" minOccurs="0"/>
               <xsd:element name="userClass" type="xsd:string"
minOccurs="0"/>
                 <xsd:element name="acSubscriberPassword"
type="xsd:string" minOccurs="0"/>
                <xsd:element name="maxDevices" type="xsd:int"
minOccurs="0"/>
                 <xsd:element name="acSubscriberStatus"
type="qipws:AC_SUBSCRIBER_STATUS_TYPE" minOccurs="0"/>
                 <xsd:element name="optionalAttributeList"
type="qipws:UdaList" minOccurs="0"/>
             </xsd:sequence>
         </xsd:extension>
     </xsd:complexContent>
```

```
    </xsd:complexType>

    <xsd:element name="AC_USERCLASS_KEY"
   type="qipws:AC_USERCLASS_KEY"/>
    <xsd:element name="AC_USERCLASS_REC"
   type="qipws:AC_USERCLASS_REC"/>
    <xsd:element name="AC_SUBSCRIBER_KEY"
   type="qipws:AC_SUBSCRIBER_KEY"/>
    <xsd:element name="AC_SUBSCRIBER_REC"
   type="qipws:AC_SUBSCRIBER_REC"/>

</xsd:schema>
```

AddressRange.xsd

```
<?xml version="1.0"?>
<!-- Copyright (c) 2005-2006 Alcatel-Lucent. -->
<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xi="http://www.w3.org/2001/XInclude"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd"/>

  <!--  define type ADDRESS_RANGE_KEYS -->
  <xsd:complexType name="ADDRESS_RANGE_KEYS">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="startAddress" type="xsd:string"
  minOccurs="0"/>
          <xsd:element name="prefixLength" type="xsd:short" minOccurs="0"/>
          <xsd:element name="name" type="xsd:string" minOccurs="0"/>
          <xsd:element name="parentBlockAddr" type="xsd:string"
  minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!--  define type ADDRESS_RANGE_REC -->
  <xsd:complexType name="ADDRESS_RANGE_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="startAddress" type="xsd:string"
  minOccurs="0"/>
          <xsd:element name="prefixLength" type="xsd:short" minOccurs="0"/>
          <xsd:element name="name" type="xsd:string" minOccurs="0"/>
          <xsd:element name="parentBlockAddr" type="xsd:string"
  minOccurs="0"/>
          <xsd:element name="description" type="xsd:string" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
```

```
  <!--  define type SEARCH_ADDRESS_RANGE_REC -->
  <xsd:complexType name="SEARCH_ADDRESS_RANGE_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="startAddress" type="xsd:string"
  minOccurs="0"/>
          <xsd:element name="prefixLength" type="xsd:short" minOccurs="0"/>
          <xsd:element name="name" type="xsd:string" minOccurs="0"/>
          <xsd:element name="parentBlockAddr" type="xsd:string"
  minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="ADDRESS_RANGE_REC" type="qipws:ADDRESS_RANGE_REC"/>
  <xsd:element name="SEARCH_ADDRESS_RANGE_REC"
  type="qipws:SEARCH_ADDRESS_RANGE_REC"/>

</xsd:schema>
```

Block.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Copyright (c) 2005-2006 Alcatel-Lucent. -->
<xsd:schema xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:xi="http://www.w3.org/2001/XInclude"
   targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
   elementFormDefault="qualified">
  <xsd:include schemaLocation="CoreTypes.xsd"/>
  <xsd:include schemaLocation="Pool.xsd"/>
    <!-- define enum AFRINIC_INETNUM_STATUS_TYPE -->
    <xsd:simpleType name="AFRINIC_INETNUM_STATUS_TYPE">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="ALLOCATED PA"/>
            <xsd:enumeration value="ALLOCATED PI"/>
            <xsd:enumeration value="ASSIGNED ANYCAST"/>
            <xsd:enumeration value="ASSIGNED PA"/>
            <xsd:enumeration value="ASSIGNED PI"/>
            <xsd:enumeration value="NOT-SET"/>
            <xsd:enumeration value="EARLY-REGISTRATION"/>
            <xsd:enumeration value="SUB-ALLOCATED PA"/>
            <xsd:enumeration value="UNSPECIFIED"/>
            <xsd:enumeration value="ALLOCATED-BY-RIR"/>
        </xsd:restriction>
    </xsd:simpleType>
    <!-- define enum AFRINIC_ORG_CONTACT_TYPE -->
    <xsd:simpleType name="AFRINIC_ORG_CONTACT_TYPE">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="AUTO-1"/>
            <xsd:enumeration value="NIC-handle"/>
        </xsd:restriction>
    </xsd:simpleType>
  <!-- define enum APNIC_INETNUM_STATUS_TYPE -->
  <xsd:simpleType name="APNIC_INETNUM_STATUS_TYPE">
     <xsd:restriction base="xsd:string">
       <xsd:enumeration value="ALLOCATED PORTABLE"/>
       <xsd:enumeration value="ALLOCATED NON-PORTABLE"/>
       <xsd:enumeration value="ASSIGNED PORTABLE"/>
       <xsd:enumeration value="ASSIGNED NON-PORTABLE"/>
     </xsd:restriction>
  </xsd:simpleType>
  <!-- define enum BLOCK_STATUS_TYPE -->
  <xsd:simpleType name="BLOCK_STATUS_TYPE">
     <xsd:restriction base="xsd:string">
```

```
      <xsd:enumeration value="ALLOCATED"/>
      <xsd:enumeration value="FREE"/>
      <xsd:enumeration value="NEXT"/>
      <xsd:enumeration value="ORIGIN"/>
      <xsd:enumeration value="PENDING"/>
      <xsd:enumeration value="RESERVED"/>
      <xsd:enumeration value="SELECTED"/>
      <xsd:enumeration value="SITE"/>
      <xsd:enumeration value="USED"/>
    </xsd:restriction>
  </xsd:simpleType>
  <!-- define enum BLOCK_STEP_TYPE -->
  <xsd:simpleType name="BLOCK_STEP_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="ALLOCATE"/>
      <xsd:enumeration value="CREATE_INFRASTRUCTURE"/>
      <xsd:enumeration value="CREATE_POOL"/>
      <xsd:enumeration value="RIR_REPORT"/>
    </xsd:restriction>
  </xsd:simpleType>
  <!-- define enum CONTACT_TYPE-->
  <xsd:simpleType name="CONTACT_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="P"/>
      <xsd:enumeration value="R"/>
    </xsd:restriction>
  </xsd:simpleType>
  <!-- define enum FREE_PENDING_CHECK_TYPE -->
  <xsd:simpleType name="FREE_PENDING_CHECK_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="BLOCK"/>
      <xsd:enumeration value="POOL"/>
      <xsd:enumeration value="ORGANIZATION"/>
      <xsd:enumeration value="ALL"/>
    </xsd:restriction>
  </xsd:simpleType>
  <!-- define enum INETNUM_ADMIN_CONTACT_TYPE -->
  <xsd:simpleType name="INETNUM_ADMIN_CONTACT_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="AUTO-2"/>
      <xsd:enumeration value="NIC-handle"/>
    </xsd:restriction>
  </xsd:simpleType>
  <!-- define enum INETNUM_ORG_TYPE -->
  <xsd:simpleType name="INETNUM_ORG_TYPE">
```

```
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="IANA"/>
      <xsd:enumeration value="LIR"/>
      <xsd:enumeration value="NIR"/>
      <xsd:enumeration value="OTHER"/>
      <xsd:enumeration value="RIR"/>
    </xsd:restriction>
  </xsd:simpleType>
  <!-- define enum INETNUM_TECH_CONTACT_TYPE -->
  <xsd:simpleType name="INETNUM_TECH_CONTACT_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="AUTO-3"/>
      <xsd:enumeration value="NIC-handle"/>
    </xsd:restriction>
  </xsd:simpleType>
    <!-- define enum NET_POC_TYPE> -->
    <xsd:simpleType name="NET_POC_TYPE">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="T"/>
            <xsd:enumeration value="AB"/>
            <xsd:enumeration value="N"/>
            <xsd:enumeration value="T_AND_AB"/>
            <xsd:enumeration value="T_AND_N"/>
            <xsd:enumeration value="AB_AND_N"/>
            <xsd:enumeration value="T_AB_AND_N"/>
        </xsd:restriction>
    </xsd:simpleType>
  <!-- define enum REGISTATION_ACTION_TYPE-->
  <xsd:simpleType name="REGISTRATION_ACTION_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="N"/>
      <xsd:enumeration value="M"/>
      <xsd:enumeration value="R"/>
    </xsd:restriction>
  </xsd:simpleType>
  <!-- define enum RIPE_INETNUM_STATUS_TYPE -->
  <xsd:simpleType name="RIPE_INETNUM_STATUS_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="ALLOCATED PA"/>
      <xsd:enumeration value="ALLOCATED PI"/>
      <xsd:enumeration value="ALLOCATED UNSPECIFIED"/>
           <xsd:enumeration value="ASSIGNED ANYCAST"/>
      <xsd:enumeration value="ASSIGNED PA"/>
      <xsd:enumeration value="ASSIGNED PI"/>
           <xsd:enumeration value="EARLY-REGISTRATION"/>
```

```
            <xsd:enumeration value="LIR-PARTITIONED PA"/>
            <xsd:enumeration value="LIR-PARTITIONED PI"/>
            <xsd:enumeration value="SUB-ALLOCATED PA"/>
            <xsd:enumeration value="NOT-SET"/>
       <!-- Above four are valid for RIPE_IPV4_INETNUM and
                following two valid for RIPE_IPV6_INETNUM only -->
            <xsd:enumeration value="ALLOCATED-BY-LIR"/>
            <xsd:enumeration value="ALLOCATED-BY-RIR"/>
            <xsd:enumeration value="ASSIGNED"/>
            <xsd:enumeration value="ASSIGNED ANYCAST"/>
    </xsd:restriction>
  </xsd:simpleType>
    <!-- define enum RIPE_ORG_CONTACT_TYPE -->
    <xsd:simpleType name="RIPE_ORG_CONTACT_TYPE">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="AUTO-1"/>
            <xsd:enumeration value="NIC-handle"/>
        </xsd:restriction>
    </xsd:simpleType>
  <!-- define enum RIR_NETWORK_NAME_OPTION_TYPE -->
  <xsd:simpleType name="RIR_NETWORK_NAME_OPTION_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="GENERATE_UNIQUE"/>
      <xsd:enumeration value="USE_POOL_NAME"/>
      <xsd:enumeration value="USER_DEFINED"/>
    </xsd:restriction>
  </xsd:simpleType>
  <!-- define enum RIR_REPORT_TYPE -->
  <xsd:simpleType name="RIR_REPORT_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="ARIN-Reallocate"/>
      <xsd:enumeration value="ARIN-Reassign-Detailed"/>
      <xsd:enumeration value="ARIN-Reassign-Simple"/>
      <xsd:enumeration value="ARIN-IPv6-Reallocate"/>
      <xsd:enumeration value="ARIN-IPv6-Reassign"/>
      <xsd:enumeration value="INETNUM for IPv4"/>
      <xsd:enumeration value="INETNUM for IPv6"/>
      <xsd:enumeration value="Do Not Generate Report"/>
    </xsd:restriction>
  </xsd:simpleType>
    <!-- Define type AFRINIC_INETNUM_REPORT -->
    <xsd:complexType name="AFRINIC_INETNUM_REPORT">
        <xsd:sequence>
            <xsd:element name="networkNameOption"
 type="qipws:RIR_NETWORK_NAME_OPTION_TYPE"/>
```

```
            <xsd:element name="descr" type="xsd:string"/>
            <xsd:element name="country" type="xsd:string"/>
            <xsd:element name="status"
type="qipws:AFRINIC_INETNUM_STATUS_TYPE"/>
            <xsd:element name="adminC"
type="qipws:INETNUM_ADMIN_CONTACT_TYPE"/>
            <xsd:element name="techC"
type="qipws:INETNUM_TECH_CONTACT_TYPE"/>
            <xsd:element name="org" type="qipws:AFRINIC_ORG_CONTACT_TYPE"
minOccurs="0"/>
            <xsd:element name="netName" type="xsd:string" minOccurs="0"/>
            <xsd:element name="orgNicHandler" type="xsd:string"
minOccurs="0"/>
            <xsd:element name="adminNicHandler" type="xsd:string"
minOccurs="0"/>
            <xsd:element name="techNicHandler" type="xsd:string"
minOccurs="0"/>
            <xsd:element name="revSrv" type="xsd:string" minOccurs="0"/>
            <xsd:element name="remarks" type="xsd:string" minOccurs="0"/>
            <xsd:element name="mntByPassword" type="xsd:string"
minOccurs="0"/>
            <xsd:element name="mntLower" type="xsd:string" minOccurs="0"/>
            <xsd:element name="mntLowerPassword" type="xsd:string"
minOccurs="0"/>
            <xsd:element name="mntDomains" type="xsd:string" minOccurs="0"/>
            <xsd:element name="mntRoutes" type="xsd:string" minOccurs="0"/>
            <xsd:element name="mntIrt" type="xsd:string" minOccurs="0"/>
            <xsd:element name="notify" type="xsd:string" minOccurs="0"/>
            <xsd:element name="changed" type="xsd:string" minOccurs="0"/>
            <xsd:element name="source" type="xsd:string" minOccurs="0"/>
            <xsd:element name="customizedInfo" type="xsd:string"
minOccurs="0"/>
            <xsd:element name="organisationObject"
type="qipws:AFRINIC_ORG_OBJECT" minOccurs="0"/>
            <xsd:element name="adminContactObject"
type="qipws:INETNUM_CONTACT_OBJECT" minOccurs="0"/>
            <xsd:element name="techContactObject"
type="qipws:INETNUM_CONTACT_OBJECT" minOccurs="0"/>
        </xsd:sequence>
    </xsd:complexType>
    <!-- Define type AFRINIC_ORG_OBJECT -->
    <xsd:complexType name="AFRINIC_ORG_OBJECT">
        <xsd:sequence>
            <xsd:element name="orgName" type="xsd:string"/>
            <xsd:element name="orgType" type="qipws:INETNUM_ORG_TYPE"/>
            <xsd:element name="address" type="xsd:string"/>
            <xsd:element name="email" type="xsd:string"/>
```

```
            <xsd:element name="country" type="xsd:string"/>
            <xsd:element name="mntRef" type="xsd:string"/>
            <xsd:element name="descr" type="xsd:string" minOccurs="0"/>
            <xsd:element name="remarks" type="xsd:string" minOccurs="0"/>
            <xsd:element name="phone" type="xsd:string" minOccurs="0"/>
            <xsd:element name="faxNo" type="xsd:string" minOccurs="0"/>
            <xsd:element name="org" type="xsd:string" minOccurs="0"/>
            <xsd:element name="refNfy" type="xsd:string" minOccurs="0"/>
            <xsd:element name="notify" type="xsd:string" minOccurs="0"/>
            <xsd:element name="abuseMailbox" type="xsd:string"
  minOccurs="0"/>
            <xsd:element name="changed" type="xsd:string" minOccurs="0"/>
            <xsd:element name="source" type="xsd:string" minOccurs="0"/>
            <xsd:element name="customizedInfo" type="xsd:string"
  minOccurs="0"/>
         </xsd:sequence>
    </xsd:complexType>
  <!-- Define type APNIC_INETNUM_REPORT -->
  <xsd:complexType name="APNIC_INETNUM_REPORT">
    <xsd:sequence>
       <xsd:element name="networkNameOption"
  type="qipws:RIR_NETWORK_NAME_OPTION_TYPE"/>
       <xsd:element name="netName" type="xsd:string" minOccurs="0"/>
       <xsd:element name="descr" type="xsd:string" minOccurs="0"/>
       <xsd:element name="country" type="xsd:string"/>
       <xsd:element name="adminC" type="qipws:INETNUM_ADMIN_CONTACT_TYPE"/>
       <xsd:element name="adminNicHandler" type="xsd:string" minOccurs="0"/>
       <xsd:element name="techC" type="qipws:INETNUM_TECH_CONTACT_TYPE"/>
       <xsd:element name="techNicHandler" type="xsd:string" minOccurs="0"/>
       <xsd:element name="revSrv" type="xsd:string" minOccurs="0"/>
       <xsd:element name="status" type="qipws:APNIC_INETNUM_STATUS_TYPE"/>
       <xsd:element name="remarks" type="xsd:string" minOccurs="0"/>
       <xsd:element name="mntByPassword" type="xsd:string" minOccurs="0"/>
       <xsd:element name="mntLower" type="xsd:string" minOccurs="0"/>
       <xsd:element name="mntLowerPassword" type="xsd:string"
  minOccurs="0"/>
       <xsd:element name="mntRoutes" type="xsd:string" minOccurs="0"/>
       <xsd:element name="notify" type="xsd:string" minOccurs="0"/>
       <xsd:element name="changed" type="xsd:string" minOccurs="0"/>
       <xsd:element name="source" type="xsd:string" minOccurs="0"/>
       <xsd:element name="customizedInfo" type="xsd:string" minOccurs="0"/>
       <xsd:element name="adminContactObject"
  type="qipws:INETNUM_CONTACT_OBJECT" minOccurs="0"/>
       <xsd:element name="techContactObject"
  type="qipws:INETNUM_CONTACT_OBJECT" minOccurs="0"/>
    </xsd:sequence>
```

```
</xsd:complexType>
<!-- define type ARIN_RA_DETAILED_REPORT -->
<xsd:complexType name="ARIN_RA_DETAILED_REPORT">
  <xsd:sequence>
    <xsd:element name="downStreamOrgId" type="xsd:string" minOccurs="0"/>
    <xsd:element name="orgInfo" type="qipws:INFO_REC" minOccurs="0"/>
    <xsd:element name="orgContactInfo" type="qipws:CONTACT_INFO_REC"
minOccurs="0"/>
    <xsd:element name="networkContactInfo"
type="qipws:CONTACT_INFO_REC"/>
    <xsd:element name="originAs" type="xsd:string" minOccurs="0"/>
    <xsd:element name="dnsReverseMappingNameserver1" type="xsd:string"
minOccurs="0"/>
    <xsd:element name="dnsReverseMappingNameserver2" type="xsd:string"
minOccurs="0"/>
    <xsd:element name="netPOCType" type="qipws:NET_POC_TYPE"
minOccurs="0"/>
    <xsd:element name="justification" type="xsd:string" minOccurs="0"/>
    <xsd:element name="networkCustomizedInfo" type="xsd:string"
minOccurs="0"/>
    <xsd:element name="publicComments" type="xsd:string" minOccurs="0"/>
    <xsd:element name="additionalInformation" type="xsd:string"
minOccurs="0"/>
    <xsd:element name="otherCustomizedInfo" type="xsd:string"
minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>
<!--  define type ARIN_RA_SIMPLE_REPORT -->
<xsd:complexType name="ARIN_RA_SIMPLE_REPORT">
  <xsd:sequence>
    <xsd:element name="registrationAction"
type="qipws:REGISTRATION_ACTION_TYPE" minOccurs="0"/>
    <xsd:element name="originAs" type="xsd:string" minOccurs="0"/>
    <xsd:element name="customerInfo" type="qipws:INFO_REC"
minOccurs="0"/>
    <xsd:element name="publicComments" type="xsd:string" minOccurs="0"/>
    <xsd:element name="customizedInfo" type="xsd:string" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>
<!--  define type CONTACT_INFO_REC  -->
<xsd:complexType name="CONTACT_INFO_REC">
  <xsd:sequence>
    <xsd:element name="handle" type="xsd:string" minOccurs="0"/>
    <xsd:element name="contactType" type="qipws:CONTACT_TYPE"
minOccurs="0"/>
    <xsd:element name="lastNameOrRoleAccount" type="xsd:string"
minOccurs="0"/>
```

```
      <xsd:element name="firstName" type="xsd:string" minOccurs="0"/>
      <xsd:element name="companyName" type="xsd:string" minOccurs="0"/>
      <xsd:element name="address" type="xsd:string" minOccurs="0"/>
      <xsd:element name="city" type="xsd:string" minOccurs="0"/>
      <xsd:element name="stateOrProvince" type="xsd:string" minOccurs="0"/>
      <xsd:element name="postalCode" type="xsd:string" minOccurs="0"/>
      <xsd:element name="countryCode" type="xsd:string" minOccurs="0"/>
      <xsd:element name="phoneNumber" type="xsd:string" minOccurs="0"/>
      <xsd:element name="emailAddress" type="xsd:string" minOccurs="0"/>
      <xsd:element name="emailAddress2" type="xsd:string" minOccurs="0"/>
      <xsd:element name="customizedInfo" type="xsd:string" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
  <!-- Define type INETNUM_CONTACT_OBJECT -->
  <xsd:complexType name="INETNUM_CONTACT_OBJECT">
    <xsd:sequence>
      <xsd:element name="person" type="xsd:string"/>
      <xsd:element name="address" type="xsd:string"/>
      <xsd:element name="phone" type="xsd:string"/>
      <xsd:element name="country" type="xsd:string" minOccurs="0"/>
      <xsd:element name="faxNo" type="xsd:string" minOccurs="0"/>
      <xsd:element name="email" type="xsd:string" minOccurs="0"/>
      <xsd:element name="remarks" type="xsd:string" minOccurs="0"/>
      <xsd:element name="notify" type="xsd:string" minOccurs="0"/>
           <xsd:element name="abuseMailbox" type="xsd:string"
  minOccurs="0"/>
      <xsd:element name="mntBy" type="xsd:string" minOccurs="0"/>
      <xsd:element name="changed" type="xsd:string" minOccurs="0"/>
      <xsd:element name="source" type="xsd:string" minOccurs="0"/>
      <xsd:element name="customizedInfo" type="xsd:string" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
  <!--  define type INFO_REC  -->
  <xsd:complexType name="INFO_REC">
    <xsd:sequence>
      <xsd:element name="fullName" type="xsd:string" minOccurs="0"/>
      <xsd:element name="address" type="xsd:string" minOccurs="0"/>
      <xsd:element name="city" type="xsd:string" minOccurs="0"/>
      <xsd:element name="stateOrProvince" type="xsd:string" minOccurs="0"/>
      <xsd:element name="postalCode" type="xsd:string" minOccurs="0"/>
      <xsd:element name="countryCode" type="xsd:string" minOccurs="0"/>
      <xsd:element name="customizedInfo" type="xsd:string" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
  <!-- Define Type PARTIAL_FAILURE_REC -->
```

```
<xsd:complexType name="PARTIAL_FAILURE_REC">
  <xsd:sequence>
    <xsd:element name="step" type="qipws:BLOCK_STEP_TYPE"/>
    <xsd:element name="failureMessage" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>
<!-- Define type RIPE_INETNUM_REPORT -->
<xsd:complexType name="RIPE_INETNUM_REPORT">
  <xsd:sequence>
    <xsd:element name="networkNameOption"
type="qipws:RIR_NETWORK_NAME_OPTION_TYPE"/>
    <xsd:element name="descr" type="xsd:string"/>
    <xsd:element name="country" type="xsd:string"/>
    <xsd:element name="status" type="qipws:RIPE_INETNUM_STATUS_TYPE"/>
    <xsd:element name="adminC" type="qipws:INETNUM_ADMIN_CONTACT_TYPE"/>
    <xsd:element name="techC" type="qipws:INETNUM_TECH_CONTACT_TYPE"/>
    <xsd:element name="org" type="qipws:RIPE_ORG_CONTACT_TYPE"
minOccurs="0"/>
    <xsd:element name="netName" type="xsd:string" minOccurs="0"/>
    <xsd:element name="orgNicHandler" type="xsd:string" minOccurs="0"/>
    <xsd:element name="adminNicHandler" type="xsd:string" minOccurs="0"/>
    <xsd:element name="techNicHandler" type="xsd:string" minOccurs="0"/>
    <xsd:element name="revSrv" type="xsd:string" minOccurs="0"/>
    <xsd:element name="remarks" type="xsd:string" minOccurs="0"/>
    <xsd:element name="mntByPassword" type="xsd:string" minOccurs="0"/>
    <xsd:element name="mntLower" type="xsd:string" minOccurs="0"/>
    <xsd:element name="mntLowerPassword" type="xsd:string"
minOccurs="0"/>
    <xsd:element name="mntDomains" type="xsd:string" minOccurs="0"/>
    <xsd:element name="mntRoutes" type="xsd:string" minOccurs="0"/>
    <xsd:element name="mntIrt" type="xsd:string" minOccurs="0"/>
    <xsd:element name="notify" type="xsd:string" minOccurs="0"/>
    <xsd:element name="changed" type="xsd:string" minOccurs="0"/>
    <xsd:element name="source" type="xsd:string" minOccurs="0"/>
    <xsd:element name="customizedInfo" type="xsd:string" minOccurs="0"/>
    <xsd:element name="organisationObject" type="qipws:RIPE_ORG_OBJECT"
minOccurs="0"/>
    <xsd:element name="adminContactObject"
type="qipws:INETNUM_CONTACT_OBJECT" minOccurs="0"/>
    <xsd:element name="techContactObject"
type="qipws:INETNUM_CONTACT_OBJECT" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>
<!--  define type V6_SEED_BLOCK_REC -->
<xsd:complexType name="V6_SEED_BLOCK_REC">
  <xsd:complexContent>
```

```
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="startAddress" type="xsd:string"
minOccurs="0"/>
          <xsd:element name="prefixLength" type="xsd:short" minOccurs="0"/>
          <xsd:element name="optionalAttributeList" type="qipws:UdaList"
minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <!--  Define Type ADDRESS_BLOCK_KEYS -->
  <xsd:complexType name="ADDRESS_BLOCK_KEYS">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="startAddress" type="xsd:string"/>
          <xsd:element name="prefixLength" type="xsd:short"/>
          <xsd:element name="addressType" type="qipws:V4V6_TYPE"
minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <!--  define type ADDRESS_BLOCK_REC -->
  <xsd:complexType name="ADDRESS_BLOCK_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="startAddress" type="xsd:string"/>
          <xsd:element name="prefixLength" type="xsd:short" minOccurs="0"/>
          <xsd:element name="addressType" type="qipws:V4V6_TYPE"
minOccurs="0"/>
          <xsd:element name="pool" type="qipws:POOL_KEYS" minOccurs="0"/>
          <xsd:element name="blockStatus" type="qipws:BLOCK_STATUS_TYPE"
minOccurs="0"/>
          <xsd:element name="parentAddressBlock"
type="qipws:ADDRESS_BLOCK_KEYS" minOccurs="0"/>
          <xsd:element name="optionalAttributeList" type="qipws:UdaList"
minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <!-- Define type RIPE_ORG_OBJECT -->
  <xsd:complexType name="RIPE_ORG_OBJECT">
```

```
    <xsd:sequence>
      <xsd:element name="orgName" type="xsd:string"/>
      <xsd:element name="orgType" type="qipws:INETNUM_ORG_TYPE"/>
      <xsd:element name="address" type="xsd:string"/>
      <xsd:element name="email" type="xsd:string"/>
      <xsd:element name="mntRef" type="xsd:string"/>
      <xsd:element name="descr" type="xsd:string" minOccurs="0"/>
      <xsd:element name="remarks" type="xsd:string" minOccurs="0"/>
      <xsd:element name="phone" type="xsd:string" minOccurs="0"/>
      <xsd:element name="faxNo" type="xsd:string" minOccurs="0"/>
      <xsd:element name="refNfy" type="xsd:string" minOccurs="0"/>
      <xsd:element name="notify" type="xsd:string" minOccurs="0"/>
           <xsd:element name="abuseMailbox" type="xsd:string"
minOccurs="0"/>
      <xsd:element name="changed" type="xsd:string" minOccurs="0"/>
      <xsd:element name="source" type="xsd:string" minOccurs="0"/>
      <xsd:element name="customizedInfo" type="xsd:string" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
  <!-- Define type RIR_REPORT -->
  <xsd:complexType name="RIR_REPORT">
    <xsd:sequence>
      <xsd:element name="reportType" type="qipws:RIR_REPORT_TYPE"/>
      <xsd:element name="arinReport" type="qipws:ARIN_RA_DETAILED_REPORT"
minOccurs="0"/>
      <xsd:element name="arinReassignSimpleReport"
type="qipws:ARIN_RA_SIMPLE_REPORT" minOccurs="0"/>
      <xsd:element name="ripeReport" type="qipws:RIPE_INETNUM_REPORT"
minOccurs="0"/>
      <xsd:element name="apnicReport" type="qipws:APNIC_INETNUM_REPORT"
minOccurs="0"/>
           <xsd:element name="afrinicReport"
type="qipws:AFRINIC_INETNUM_REPORT" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
  <!--  define type SEARCH_ADDRESS_BLOCK_REC -->
  <xsd:complexType name="SEARCH_ADDRESS_BLOCK_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="addressType" type="qipws:V4V6_TYPE"
minOccurs="0"/>
          <xsd:element name="startAddress" type="xsd:string"
minOccurs="0"/>
          <xsd:element name="prefixLength" type="xsd:short" minOccurs="0"/>
          <xsd:element name="poolName" type="xsd:string" minOccurs="0"/>
```

```
          <xsd:element name="blockStatusList"
type="qipws:BLOCK_STATUS_TYPE" minOccurs="0" maxOccurs="unbounded"/>
          <xsd:element name="registry" type="qipws:REGISTRY_TYPE"
minOccurs="0"/>
          <xsd:element name="poolOptionalAttributeList"
type="qipws:UdaList" minOccurs="0"/>
          <xsd:element name="blockOptionalAttributeList"
type="qipws:UdaList" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <!--  define type SEED_BLOCK_REC -->
  <xsd:complexType name="SEED_BLOCK_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:ADDRESS_BLOCK_REC">
        <xsd:sequence>
          <xsd:element name="maintainer" type="xsd:string" minOccurs="0"/>
          <xsd:element name="threshold" type="xsd:short" minOccurs="0"/>
          <xsd:element name="reportTypes" type="qipws:RIR_REPORT_TYPE"
minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <!-- define element AddSeedBlockToSeedPoolRequest data -->
  <xsd:element name="AddSeedBlockToSeedPoolRequest">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="qipws:VQIP_BASE_REQUEST">
          <xsd:sequence>
            <xsd:element name="addressBlock" type="qipws:SEED_BLOCK_REC"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <!-- define element AllocateBlockToPoolMessageRequest data -->
  <xsd:element name="AllocateBlockToPoolMessageRequest">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="qipws:VQIP_BASE_REQUEST">
          <xsd:sequence>
            <xsd:element name="parameters"
type="qipws:AllocateBlockToPoolRequest"/>
          </xsd:sequence>
```

```
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <!--  define type AllocateBlockToPoolRequest -->
  <xsd:complexType name="AllocateBlockToPoolRequest">
    <xsd:sequence>
      <xsd:element name="pool" type="qipws:POOL_KEYS"/>
      <xsd:element name="addressType" type="qipws:V4V6_TYPE"/>
      <xsd:element name="requestedBlock" type="qipws:ADDRESS_BLOCK_KEYS"
  minOccurs="0"/>
      <xsd:element name="networkContact" type="xsd:string" minOccurs="0"/>
      <xsd:element name="ruleName" type="xsd:string" minOccurs="0"/>
      <xsd:element name="subnetName" type="xsd:string" minOccurs="0"/>
      <xsd:element name="optionalAttributeList" type="qipws:UdaList"
  minOccurs="0"/>
      <xsd:element name="rirReport" type="qipws:RIR_REPORT" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:element name="AllocateBlockToPoolRequest"
  type="qipws:AllocateBlockToPoolRequest" />
  <!-- define block allocation response data -->
  <xsd:element name="BlockAllocResponse">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="qipws:VQIP_BASE_RESPONSE">
          <xsd:sequence>
            <xsd:element name="errorRecs" type="qipws:PARTIAL_FAILURE_REC"
  minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="pool" type="qipws:POOL_KEYS" />
                 <xsd:element name="block" type="qipws:ADDRESS_BLOCK_KEYS"
  />
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <!-- define element DeleteSeedBlockFromSeedPoolRequest data -->
  <xsd:element name="DeleteSeedBlockFromSeedPoolRequest">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="qipws:VQIP_BASE_REQUEST">
          <xsd:sequence>
            <xsd:element name="addressBlock"
  type="qipws:ADDRESS_BLOCK_REC"/>
          </xsd:sequence>
```

```
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <!-- define element ExpandBlockRequest data -->
  <xsd:element name="ExpandBlockRequest">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="qipws:VQIP_BASE_REQUEST">
          <xsd:sequence>
            <xsd:element name="addressBlock"
  type="qipws:ADDRESS_BLOCK_REC"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <!-- define element ExplicitBlockAllocMessageRequest data -->
  <xsd:element name="ExplicitBlockAllocMessageRequest">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="qipws:VQIP_BASE_REQUEST">
          <xsd:sequence>
            <xsd:element name="parameters"
  type="qipws:ExplicitBlockAllocRequest"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <!--  define type ExplicitBlockAllocRequest -->
  <xsd:complexType name="ExplicitBlockAllocRequest">
    <xsd:sequence>
      <xsd:element name="startAddress" type="xsd:string"/>
      <xsd:element name="prefixLength" type="xsd:short"/>
      <xsd:element name="pool" type="qipws:POOL_KEYS"/>
      <xsd:element name="blockStatus" type="qipws:BLOCK_STATUS_TYPE"/>
      <xsd:element name="addressType" type="qipws:V4V6_TYPE"
  minOccurs="0"/>
      <xsd:element name="subnetCreationOption"
  type="qipws:SUBNET_CREATION_OPTION_TYPE" minOccurs="0"/>
      <xsd:element name="addressTemplate" type="xsd:string" minOccurs="0"/>
      <xsd:element name="subnetName" type="xsd:string" minOccurs="0"/>
      <xsd:element name="subnetProfileTemplateName" type="xsd:string"
  minOccurs="0"/>
      <xsd:element name="networkContact" type="xsd:string" minOccurs="0"/>
```

```
      <xsd:element name="allocationRuleName" type="xsd:string"
minOccurs="0"/>
      <xsd:element name="useDefaultRuleSize" type="xsd:boolean"
minOccurs="0"/>
      <xsd:element name="optionalAttributeList" type="qipws:UdaList"
minOccurs="0"/>
      <xsd:element name="rirReport" type="qipws:RIR_REPORT" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:element name="ExplicitBlockAllocRequest"
  type="qipws:ExplicitBlockAllocRequest" />
  <!-- define element FreeBlockRequest data -->
  <xsd:element name="FreeBlockRequest">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="qipws:VQIP_BASE_REQUEST">
          <xsd:sequence>
            <xsd:element name="block" type="qipws:ADDRESS_BLOCK_REC"/>
            <xsd:element name="returnToParent" type="xsd:boolean"
minOccurs="0"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <!-- define element FreePendingBlockRequest data -->
  <xsd:element name="FreePendingBlockRequest">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="qipws:VQIP_BASE_REQUEST">
          <xsd:sequence>
            <xsd:element name="level"
type="qipws:FREE_PENDING_CHECK_TYPE"/>
            <xsd:element name="override" type="xsd:boolean" minOccurs="0"/>
            <xsd:element name="block" type="qipws:ADDRESS_BLOCK_KEYS"
minOccurs="0"/>
            <xsd:element name="pool" type="qipws:POOL_KEYS" minOccurs="0"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <!-- define element QuickBlockAllocMessageRequest data -->
  <xsd:element name="QuickBlockAllocMessageRequest">
    <xsd:complexType>
      <xsd:complexContent>
```

```
        <xsd:extension base="qipws:VQIP_BASE_REQUEST">
          <xsd:sequence>
            <xsd:element name="parameters"
type="qipws:QuickBlockAllocRequest"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <!--  define type QuickBlockAllocRequest -->
  <xsd:complexType name="QuickBlockAllocRequest">
    <xsd:sequence>
      <xsd:element name="parentPool" type="qipws:POOL_KEYS"/>
      <xsd:element name="childPoolName" type="xsd:string"/>
      <xsd:element name="addressType" type="qipws:V4V6_TYPE"/>
      <xsd:element name="ruleName" type="xsd:string"/>
      <xsd:element name="poolContact" type="xsd:string" minOccurs="0"/>
      <xsd:element name="networkContact" type="xsd:string" minOccurs="0"/>
      <xsd:element name="subnetName" type="xsd:string" minOccurs="0"/>
      <xsd:element name="childPoolOptionalAttributeList"
type="qipws:UdaList" minOccurs="0"/>
      <xsd:element name="addressBlockOptionalAttributeList"
type="qipws:UdaList" minOccurs="0"/>
      <xsd:element name="rirReport" type="qipws:RIR_REPORT" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:element name="QuickBlockAllocRequest"
  type="qipws:QuickBlockAllocRequest" />
  <!-- define element RecoverBlockRequest data -->
  <xsd:element name="RecoverBlockRequest">
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="qipws:VQIP_BASE_REQUEST">
          <xsd:sequence>
            <xsd:element name="block" type="qipws:ADDRESS_BLOCK_KEYS"/>
            <xsd:element name="networkContact" type="xsd:string"
minOccurs="0"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <!-- define recover block response data -->
  <xsd:element name="RecoverBlockResponse">
    <xsd:complexType>
      <xsd:complexContent>
```

```
          <xsd:extension base="qipws:VQIP_BASE_RESPONSE">
            <xsd:sequence>
              <xsd:element name="blockKey" type="qipws:ADDRESS_BLOCK_KEYS"/>
            </xsd:sequence>
          </xsd:extension>
        </xsd:complexContent>
      </xsd:complexType>
    </xsd:element>
    <!-- define element ReturnToParentBlockRequest data -->
    <xsd:element name="ReturnToParentBlockRequest">
      <xsd:complexType>
        <xsd:complexContent>
          <xsd:extension base="qipws:VQIP_BASE_REQUEST">
            <xsd:sequence>
              <xsd:element name="addressBlock"
    type="qipws:ADDRESS_BLOCK_REC"/>
            </xsd:sequence>
          </xsd:extension>
        </xsd:complexContent>
      </xsd:complexType>
    </xsd:element>

    <xsd:element name="ADDRESS_BLOCK_REC" type="qipws:ADDRESS_BLOCK_REC"/>
    <xsd:element name="SEED_BLOCK_REC" type="qipws:SEED_BLOCK_REC"/>
    <xsd:element name="V6_SEED_BLOCK_REC" type="qipws:V6_SEED_BLOCK_REC"/>
    <xsd:element name="SEARCH_ADDRESS_BLOCK_REC"
    type="qipws:SEARCH_ADDRESS_BLOCK_REC"/>
</xsd:schema>
```

CoreTypes.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Copyright (c) 2005-2009 Alcatel-Lucent. -->
<xsd:schema targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xi="http://www.w3.org/2001/XInclude"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:qipws="http://alcatel-
   lucent.com/qip/nb/ws" elementFormDefault="qualified">
    <!-- define enum ACTION_TYPE -->
    <xsd:simpleType name="ACTION_TYPE">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="A"/>
            <xsd:enumeration value="S"/>
            <xsd:enumeration value="D"/>
            <xsd:enumeration value="U"/>
            <xsd:enumeration value="G"/>
            <xsd:enumeration value="C"/>
        </xsd:restriction>
    </xsd:simpleType>
    <!-- define enum ADDRESS_TYPE -->
    <xsd:simpleType name="ADDRESS_TYPE">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="STATIC"/>
            <xsd:enumeration value="DYNAMIC"/>
            <xsd:enumeration value="RESERVED"/>
        </xsd:restriction>
    </xsd:simpleType>
    <!-- define enum IPADDR_ALLOC_ALG_TYPE -->
    <xsd:simpleType name="IPADDR_ALLOC_ALG_TYPE">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="NEXT_AVAILABLE"/>
            <xsd:enumeration value="LAST_AVAILABLE"/>
            <xsd:enumeration value="RANDOM"/>
            <xsd:enumeration value="SPECIFIC"/>
            <xsd:enumeration value="EXISTING"/>
        </xsd:restriction>
    </xsd:simpleType>
    <!-- define enum ENTITY_TYPE -->
    <xsd:simpleType name="ENTITY_TYPE">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="CHILD_POOL"/>
        </xsd:restriction>
    </xsd:simpleType>
    <!-- define enum HW_TYPE -->
    <xsd:simpleType name="HW_TYPE">
```

```
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="ETHERNET"/>
            <xsd:enumeration value="TOKEN_RING"/>
            <xsd:enumeration value="AX25"/>
            <xsd:enumeration value="PRONET"/>
            <xsd:enumeration value="CHOAS"/>
            <xsd:enumeration value="IEEE802"/>
            <xsd:enumeration value="ARCNET"/>
        </xsd:restriction>
    </xsd:simpleType>
    <!-- define enum LIST_TYPE -->
    <xsd:simpleType name="LIST_TYPE">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="DHCP_SRV"/>
            <xsd:enumeration value="DHCP_OPT_TEMPL"/>
            <xsd:enumeration value="DHCP_SCOPE_POL_TEMPL"/>
            <xsd:enumeration value="DNS_SRV"/>
            <xsd:enumeration value="DNS_VIEWS"/>
            <xsd:enumeration value="DOMAIN"/>
            <xsd:enumeration value="OBJ_CLASS"/>
            <xsd:enumeration value="SUBNET_ORGANIZATION"/>
            <xsd:enumeration value="UDA_PRE_EDIT_CALLOUT"/>
            <xsd:enumeration value="UDA_POST_EDIT_CALLOUT"/>
            <xsd:enumeration value="UDA_VALIDATION_CALLOUT"/>
            <xsd:enumeration value="USER_CLASS"/>
            <xsd:enumeration value="VENDOR_CLASS"/>
        </xsd:restriction>
    </xsd:simpleType>
    <!-- define enum OS_TYPE -->
    <xsd:simpleType name="OS_TYPE">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="WINDOWS"/>
            <xsd:enumeration value="UNIX"/>
        </xsd:restriction>
    </xsd:simpleType>
    <!-- define enum REGISTRY_TYPE -->
    <xsd:simpleType name="REGISTRY_TYPE">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="ARIN"/>
            <xsd:enumeration value="RIPE"/>
            <xsd:enumeration value="APNIC"/>
            <xsd:enumeration value="AFRINIC"/>
        </xsd:restriction>
    </xsd:simpleType>
    <!-- define enum RESULT_TYPE -->
```

```
    <xsd:simpleType name="RESULT_TYPE">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="UNKNOWN"/>
            <xsd:enumeration value="SUCCESS"/>
            <xsd:enumeration value="NOT_FOUND"/>
            <xsd:enumeration value="PARTIAL_FAILURE"/>
            <!-- NOT_FOUND applicable for delete operation -->
        </xsd:restriction>
    </xsd:simpleType>
    <!-- define enum SCHEDULE_JOB_TYPE -->
    <xsd:simpleType name="SCHEDULE_JOB_TYPE">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="IN_FOREGROUND"/>
            <xsd:enumeration value="IN_BACKGROUND"/>
            <xsd:enumeration value="SCHEDULED_AT"/>
        </xsd:restriction>
    </xsd:simpleType>
    <!-- define enum START_TYPE -->
    <xsd:simpleType name="START_TYPE">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="AT"/>
            <xsd:enumeration value="AFTER"/>
        </xsd:restriction>
    </xsd:simpleType>
    <!-- define enum SUBNET_CREATION_OPTION_TYPE -->
    <xsd:simpleType name="SUBNET_CREATION_OPTION_TYPE">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="CREATE_NEW"/>
            <xsd:enumeration value="LINK_TO_EXISTING"/>
            <xsd:enumeration value="CREATE_NEW_IF_NECESSARY"/>
            <xsd:enumeration value="DO_NOT_CREATE_OR_LINK"/>
        </xsd:restriction>
    </xsd:simpleType>
    <!-- define enum V4V6_TYPE -->
    <xsd:simpleType name="V4V6_TYPE">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="IPv4"/>
            <xsd:enumeration value="IPv6"/>
        </xsd:restriction>
    </xsd:simpleType>
    <!-- define type CommonInfo -->
    <xsd:complexType name="CommonInfo">
        <xsd:sequence>
            <xsd:element name="organization" type="xsd:string"
minOccurs="0"/>
```

```
            <xsd:element name="orgID" type="xsd:int" minOccurs="0"/>
            <xsd:element name="locale" type="xsd:string" minOccurs="0"/>
            <xsd:element name="site" type="xsd:string" minOccurs="0"/>
        </xsd:sequence>
    </xsd:complexType>
    <!-- define element CommonInfo used by all requests -->
    <xsd:element name="CommonInfo" type="qipws:CommonInfo"/>
    <!-- define element GetAllRequest -->
    <xsd:element name="GetAllRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="listType" type="qipws:LIST_TYPE"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element GetAllResponse -->
    <xsd:element name="GetAllResponse">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_RESPONSE">
                    <xsd:sequence>
                        <xsd:element name="list" type="qipws:LIST_REC"
minOccurs="0"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element GetAssociatedComponentsRequest -->
    <xsd:element name="GetAssociatedComponentsRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="param"
type="qipws:VQIP_KEY_ENTITY"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
```

```
    <!-- define element GetAssociatedComponentsResponse -->
    <xsd:element name="GetAssociatedComponentsResponse">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_RESPONSE">
                    <xsd:sequence>
                        <xsd:element name="items"
  type="qipws:LIST_BASE_ENTITY"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element GetFileRequest -->
    <xsd:element name="GetFileRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="reqObject"
  type="qipws:VQIP_KEY_ENTITY"/>
                        <xsd:element name="os" type="qipws:OS_TYPE"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define type CONTACT_REC -->
    <xsd:complexType name="CONTACT_REC">
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_BASE_ENTITY">
                <xsd:sequence>
                    <xsd:element name="firstName" type="xsd:string"
  minOccurs="0"/>
                    <xsd:element name="lastName" type="xsd:string"
  minOccurs="0"/>
                    <xsd:element name="telephone" type="xsd:string"
  minOccurs="0"/>
                    <xsd:element name="pager" type="xsd:string"
  minOccurs="0"/>
                    <xsd:element name="email" type="xsd:string"
  minOccurs="0"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>
```

```
    <!-- define element CopyRuleRequest -->
    <xsd:element name="CopyRuleRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="fromRule"
type="qipws:VQIP_BASE_ENTITY"/>
                        <xsd:element name="toRule" type="xsd:string"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element AddContainedObjectRequest -->
    <xsd:element name="AddContainedObjectRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="parentKey"
type="qipws:VQIP_KEY_ENTITY"/>
                        <xsd:element name="containedObject"
type="qipws:VQIP_BASE_ENTITY"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element CancelRequest -->
    <xsd:element name="CancelRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="reqObject"
type="qipws:VQIP_BASE_ENTITY"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element CopyRequest -->
    <xsd:element name="CopyRequest">
        <xsd:complexType>
            <xsd:complexContent>
```

```
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                      <xsd:element name="from" type="qipws:VQIP_KEY_ENTITY"/>
                        <xsd:element name="to" type="qipws:VQIP_KEY_ENTITY"/>
                       <xsd:element name="addlParams" type="qipws:NameValue"
minOccurs="0" maxOccurs="unbounded"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element CountAssociationsRequest -->
    <xsd:element name="CountAssociationsRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="objectKey"
type="qipws:VQIP_KEY_ENTITY"/>
                        <xsd:element name="assocWithKey"
type="qipws:VQIP_KEY_ENTITY" minOccurs="0"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element CountAssociationsResponse -->
    <xsd:element name="CountAssociationsResponse">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_RESPONSE">
                    <xsd:sequence>
                        <xsd:element name="count" type="xsd:int"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element DeleteContainedObjectRequest -->
    <xsd:element name="DeleteContainedObjectRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="objectKey"
type="qipws:VQIP_KEY_ENTITY"/>
```

```
                    <!-- Flag to indicate if underlying objects should be
deleted
                         for example - for interfaces - to check if DNS
records
                         should be deleted.
                     -->
                     <xsd:element name="deleteRelatedRecords"
type="xsd:boolean"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>
 </xsd:element>
 <!-- define element DeleteForcedRequest -->
 <xsd:element name="DeleteForcedRequest">
    <xsd:complexType>
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                <xsd:sequence>
                    <xsd:element name="objectKey"
type="qipws:VQIP_KEY_ENTITY"/>
                  <!-- Flag to indicate if delete should be forced; even
if object is in use. -->
                    <xsd:element name="forced" type="xsd:boolean"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>
 </xsd:element>
 <!-- define element DeleteRequest -->
 <xsd:element name="DeleteRequest">
    <xsd:complexType>
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                <xsd:sequence>
                    <xsd:element name="reqObject"
type="qipws:VQIP_BASE_ENTITY"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>
 </xsd:element>
 <!-- define element FindAssociationsRequest -->
 <xsd:element name="FindAssociationsRequest">
    <xsd:complexType>
        <xsd:complexContent>
```

```
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="objectKey"
type="qipws:VQIP_KEY_ENTITY"/>
                        <xsd:element name="assocWithKey"
type="qipws:VQIP_KEY_ENTITY" minOccurs="0"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element FindAssociationsResponse -->
    <xsd:element name="FindAssociationsResponse">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_RESPONSE">
                    <xsd:sequence>
                        <xsd:element name="objects"
type="qipws:VQIP_KEY_ENTITY" minOccurs="0" maxOccurs="unbounded"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element GetRequest -->
    <xsd:element name="GetRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="reqObject"
type="qipws:VQIP_BASE_ENTITY"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element MergeRequest -->
    <xsd:element name="MergeRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="from"
type="qipws:VQIP_BASE_ENTITY"/>
                        <xsd:element name="to" type="qipws:VQIP_BASE_ENTITY"/>
```

```
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define type LIST_BASE_ENTITY -->
    <xsd:complexType name="LIST_BASE_ENTITY">
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_BASE_ENTITY"/>
        </xsd:complexContent>
    </xsd:complexType>
    <!-- define type LIST_REC -->
    <xsd:complexType name="LIST_REC">
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_BASE_ENTITY">
                <xsd:sequence>
                    <xsd:element name="names" type="xsd:string"
maxOccurs="unbounded"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>
    <!-- define element LIST_REC -->
    <xsd:element name="LIST_REC" type="qipws:LIST_REC"/>
    <!-- define type LOCATION_REC -->
    <xsd:complexType name="LOCATION_REC">
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_BASE_ENTITY">
                <xsd:sequence>
                    <xsd:element name="street1" type="xsd:string"
minOccurs="0"/>
                    <xsd:element name="street2" type="xsd:string"
minOccurs="0"/>
                   <xsd:element name="city" type="xsd:string" minOccurs="0"/>
                    <xsd:element name="state" type="xsd:string"
minOccurs="0"/>
                   <xsd:element name="zip" type="xsd:string" minOccurs="0"/>
                    <xsd:element name="country" type="xsd:string"
minOccurs="0"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>
    <!-- define enum MODIFY_RRS_TYPE -->
    <xsd:simpleType name="MODIFY_RRS_TYPE">
        <xsd:restriction base="xsd:string">
```

```
            <xsd:enumeration value="YES"/>
            <xsd:enumeration value="NO"/>
            <xsd:enumeration value="SAME_AS_GLOBAL_POLICY"/>
        </xsd:restriction>
    </xsd:simpleType>
    <!-- define type NameValue - for handling User Defined Fields -->
    <xsd:complexType name="NameValue">
        <xsd:sequence>
            <xsd:element name="name" type="xsd:string"/>
            <xsd:element name="value" type="xsd:string" minOccurs="0"/>
        </xsd:sequence>
    </xsd:complexType>
    <!-- define type StringList -->
    <xsd:complexType name="StringList">
        <xsd:sequence>
            <xsd:element name="name" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
    </xsd:complexType>
    <!-- define type REGISTRY_REC -->
    <xsd:complexType name="REGISTRY_REC">
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_BASE_ENTITY">
                <xsd:sequence>
                    <xsd:element name="name" type="qipws:REGISTRY_TYPE"
minOccurs="0"/>
                    <xsd:element name="emailAddress" type="xsd:string"
minOccurs="0"/>
                    <xsd:element name="ccList" type="xsd:string"
minOccurs="0"/>
                    <xsd:element name="minIPv4BlockSize" type="xsd:short"
minOccurs="0"/>
                    <xsd:element name="minIPv6BlockSize" type="xsd:short"
minOccurs="0"/>
                    <xsd:element name="whoisServerName" type="xsd:string"
minOccurs="0"/>
                    <xsd:element name="source" type="xsd:string"
minOccurs="0"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>
    <!-- define element REGISTRY_REC -->
    <xsd:element name="REGISTRY_REC" type="qipws:REGISTRY_REC"/>
    <!-- define element Response -->
    <xsd:element name="Response">
```

```
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_RESPONSE">
                    <xsd:sequence>
                        <xsd:element name="responseData"
type="qipws:RESPONSE_DATA" minOccurs="0"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define type RESPONSE_DATA -->
    <xsd:complexType name="RESPONSE_DATA">
        <xsd:sequence>
            <xsd:element name="jobId" type="xsd:int" minOccurs="0"/>
            <xsd:element name="name" type="xsd:string" minOccurs="0"/>
            <xsd:element name="completionTime" type="xsd:string"
minOccurs="0"/>
        </xsd:sequence>
    </xsd:complexType>
    <!-- define type ROUTER_REC -->
    <xsd:complexType name="ROUTER_REC">
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_BASE_ENTITY">
                <xsd:sequence>
                   <xsd:element name="name" type="xsd:string" minOccurs="0"/>
                    <xsd:element name="address" type="xsd:string"
minOccurs="0"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>
    <!-- define element SearchResponse -->
    <xsd:element name="SearchResponse">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_RESPONSE">
                    <xsd:sequence>
                        <xsd:element name="objectData"
type="qipws:VQIP_BASE_ENTITY" minOccurs="0" maxOccurs="unbounded"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define type JOB_REQUEST -->
```

```
  <xsd:complexType name="JOB_REQUEST">
      <xsd:sequence>
          <xsd:element name="when" type="xsd:string" minOccurs="0"/>
          <xsd:element name="toEmail" type="xsd:string" minOccurs="0"/>
          <xsd:element name="ccEmail" type="xsd:string" minOccurs="0"/>
          <xsd:element name="sendEmail" type="xsd:boolean" minOccurs="0"/>
          <xsd:element name="runMode" type="qipws:SCHEDULE_JOB_TYPE"
minOccurs="0"/>
         <xsd:element name="description" type="xsd:string" minOccurs="0"/>
      </xsd:sequence>
  </xsd:complexType>
  <!-- Define base entity type.  Though it should be declared "abstract",
not all tools yet support it. -->
  <xsd:complexType name="VQIP_BASE_ENTITY">
      <xsd:sequence/>
  </xsd:complexType>
  <!-- Define base key type.  Though it should be declared "abstract", not
all tools yet support it. -->
  <xsd:complexType name="VQIP_KEY_ENTITY">
      <xsd:sequence/>
  </xsd:complexType>
 <!-- Define base response type.  Though it should be declared "abstract",
not all tools yet support it. -->
  <xsd:complexType name="VQIP_BASE_RESPONSE">
      <xsd:sequence>
          <xsd:element name="result" type="qipws:RESULT_TYPE"/>
      </xsd:sequence>
  </xsd:complexType>
  <!-- Define base request type.  Though it should be declared "abstract",
not all tools yet support it. -->
  <xsd:complexType name="VQIP_BASE_REQUEST">
      <xsd:sequence>
          <xsd:element name="commonParam" type="qipws:CommonInfo"/>
          <xsd:element name="jobReq" type="qipws:JOB_REQUEST"
minOccurs="0"/>
      </xsd:sequence>
  </xsd:complexType>
  <!-- define type UdaGroup - for optionally grouping a list of UDAs. -->
  <xsd:complexType name="UdaGroup">
      <xsd:sequence>
          <xsd:element name="name" type="xsd:string"/>
          <xsd:element name="udas" type="qipws:NameValue"
maxOccurs="unbounded"/>
          <xsd:element name="searchType" type="qipws:UDA_SEARCH_TYPE"
minOccurs="0"/>
      </xsd:sequence>
```

```
  </xsd:complexType>
  <!-- define type UdaList -->
  <xsd:complexType name="UdaList">
      <xsd:sequence>
          <xsd:element name="udas" type="qipws:NameValue" minOccurs="0"
maxOccurs="unbounded"/>
          <xsd:element name="groups" type="qipws:UdaGroup" minOccurs="0"
maxOccurs="unbounded"/>
      </xsd:sequence>
  </xsd:complexType>
  <!-- define type UdfList -->
  <xsd:complexType name="UdfList">
      <xsd:sequence>
          <xsd:element name="udf" type="qipws:NameValue" minOccurs="0"
maxOccurs="unbounded"/>
      </xsd:sequence>
  </xsd:complexType>
  <!-- define enum UDA_SEARCH_TYPE -->
  <xsd:simpleType name="UDA_SEARCH_TYPE">
      <xsd:restriction base="xsd:string">
          <xsd:enumeration value="_ANY_"/>
          <xsd:enumeration value="_IGNORE_"/>
          <xsd:enumeration value="_NONE_"/>
          <xsd:enumeration value="_SPECIFIC_"/>
      </xsd:restriction>
  </xsd:simpleType>
  <!-- define element AddRequest -->
  <xsd:element name="AddRequest">
      <xsd:complexType>
          <xsd:complexContent>
              <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                  <xsd:sequence>
                      <xsd:element name="reqObject"
type="qipws:VQIP_BASE_ENTITY"/>
                  </xsd:sequence>
              </xsd:extension>
          </xsd:complexContent>
      </xsd:complexType>
  </xsd:element>
  <!-- define element ApplFault -->
  <xsd:element name="ApplFault">
      <xsd:complexType>
          <xsd:sequence>
              <xsd:element name="errorMsg" type="xsd:string"/>
              <xsd:element name="errorKey" type="xsd:string"/>
          </xsd:sequence>
```

```
        </xsd:complexType>
    </xsd:element>
    <!-- define element AssociateRequest -->
    <xsd:element name="AssociateRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="srcKey"
type="qipws:VQIP_KEY_ENTITY"/>
                        <xsd:element name="targetKey"
type="qipws:VQIP_KEY_ENTITY"/>
                        <xsd:element name="targetParameters"
type="qipws:VQIP_BASE_ENTITY" minOccurs="0"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element AssociateListRequest -->
    <xsd:element name="AssociateListRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="srcKeys"
type="qipws:VQIP_KEY_ENTITY" maxOccurs="unbounded"/>
                        <xsd:element name="targetKey"
type="qipws:VQIP_KEY_ENTITY"/>
                        <xsd:element name="targetParameters"
type="qipws:VQIP_BASE_ENTITY" minOccurs="0"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element DissociateRequest -->
    <xsd:element name="DissociateRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="srcKey"
type="qipws:VQIP_KEY_ENTITY"/>
                        <xsd:element name="targetKey"
type="qipws:VQIP_KEY_ENTITY"/>
```

```
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element DissociateListRequest -->
    <xsd:element name="DissociateListRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                     <xsd:element name="srcKey" type="qipws:VQIP_KEY_ENTITY"
maxOccurs="unbounded"/>
                        <xsd:element name="targetKey"
type="qipws:VQIP_KEY_ENTITY"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element GetResponsee -->
    <xsd:element name="GetResponse">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_RESPONSE">
                    <xsd:sequence>
                        <xsd:element name="record"
type="qipws:VQIP_BASE_ENTITY" minOccurs="0"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element GetContainedObjectsRequest -->
    <xsd:element name="GetContainedObjectsRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="parent"
type="qipws:VQIP_KEY_ENTITY"/>
                        <xsd:element name="containedObjectType"
type="qipws:ENTITY_TYPE"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
```

```
        </xsd:complexType>
    </xsd:element>
    <!-- define element GetContainedObjectsResponse -->
    <xsd:element name="GetContainedObjectsResponse">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_RESPONSE">
                    <xsd:sequence>
                        <xsd:element name="containedObjects"
type="qipws:VQIP_KEY_ENTITY" maxOccurs="unbounded"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element MoveRequest -->
    <xsd:element name="MoveRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="source"
type="qipws:VQIP_BASE_ENTITY"/>
                        <xsd:element name="target"
type="qipws:VQIP_BASE_ENTITY"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element PageSearchRequest -->
    <xsd:element name="PageSearchRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="reqObject"
type="qipws:VQIP_BASE_ENTITY"/>
                        <xsd:element name="pageCount" type="xsd:int"
minOccurs="0"/>
                        <xsd:element name="pageSize" type="xsd:int"
minOccurs="0"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
```

```
    </xsd:element>
    <!-- define element PageSearchResponse -->
    <xsd:element name="PageSearchResponse">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_RESPONSE">
                    <xsd:sequence>
                        <xsd:element name="objectData"
type="qipws:VQIP_BASE_ENTITY" minOccurs="0" maxOccurs="unbounded"/>
                        <xsd:element name="totalPages" type="xsd:int"/>
                        <xsd:element name="currentPage" type="xsd:int"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element RenumberRequest -->
    <xsd:element name="RenumberRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="source"
type="qipws:VQIP_BASE_ENTITY"/>
                        <xsd:element name="target"
type="qipws:VQIP_BASE_ENTITY"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element SearchRequest -->
    <xsd:element name="SearchRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="reqObject"
type="qipws:VQIP_BASE_ENTITY"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element SequenceRequest -->
    <xsd:element name="SequenceRequest">
```

```
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="reqObject"
type="qipws:VQIP_KEY_ENTITY" maxOccurs="unbounded"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element SequenceAssociatedComponentsRequest -->
    <xsd:element name="SequenceAssociatedComponentsRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                       <xsd:element name="key" type="qipws:VQIP_KEY_ENTITY"/>
                        <xsd:element name="components"
type="qipws:LIST_BASE_ENTITY"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element SplitRequest -->
    <xsd:element name="SplitRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="entity"
type="qipws:VQIP_BASE_ENTITY"/>
                        <xsd:element name="requestedSize" type="xsd:short"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
    <!-- define element UpdateRequest - generic Update request complex type
is inline and inherits from base request-->
    <xsd:element name="UpdateRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
```

```
                        <xsd:element name="reqObject"
   type="qipws:VQIP_BASE_ENTITY"/>
                       <xsd:element name="key" type="qipws:VQIP_KEY_ENTITY"
   minOccurs="0"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
     </xsd:element>
</xsd:schema>
```

DnsAcl.xsd

```
<?xml version="1.0"?>
<!-- Inserted into VQIPTypes.xsd via an XInclude. -->
<!-- Copyright (c) 2007-2008 Alcatel-Lucent. -->

<xsd:schema
     xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema"
     targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
     elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd"/>

  <!-- Define a DNS_ACL_KEY type -->
  <xsd:complexType name="DNS_ACL_TEMPLATE_KEY">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- Define a SEARCH_DNS_ACL_TEMPLATE_REC -->
  <xsd:complexType name="SEARCH_DNS_ACL_TEMPLATE_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- Define Access Type enums for "acl_other" -->
  <xsd:simpleType name="DNS_ACT_OTHER_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="ANY"/>
      <xsd:enumeration value="NONE"/>
      <xsd:enumeration value="LOCALHOST"/>
      <xsd:enumeration value="!LOCALHOST"/>
      <xsd:enumeration value="LOCALNETS"/>
      <xsd:enumeration value="!LOCALNETS"/>
```

```
    </xsd:restriction>
  </xsd:simpleType>

  <!-- Define a DNS Access Control List. -->
  <xsd:complexType name="DNS_ACL_TYPE">
    <xsd:choice minOccurs="0" maxOccurs="unbounded">
      <xsd:element name="ipv4_address" type="xsd:string" />
      <xsd:element name="ipv4_network" type="xsd:string" />
      <xsd:element name="ipv6_address" type="xsd:string" />
      <xsd:element name="ipv6_network" type="xsd:string" />
      <xsd:element name="template" type="xsd:string" />
      <xsd:element name="key" type="xsd:string" />
      <xsd:element name="other" type="qipws:DNS_ACT_OTHER_TYPE" />
      <xsd:element name="text" type="xsd:string" />
    </xsd:choice>
  </xsd:complexType>

  <!-- Define a DNS_ACL_TEMPLATE_REC as a type -->
  <xsd:complexType name="DNS_ACL_TEMPLATE_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string" minOccurs="0"/>
          <xsd:element name="acl" type="qipws:DNS_ACL_TYPE" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- Define DNS_ACL_TEMPLATE_REC as an element you can use. -->
  <xsd:element name="DNS_ACL_TEMPLATE_REC"
  type="qipws:DNS_ACL_TEMPLATE_REC"/>
  <xsd:element name="SEARCH_DNS_ACL_TEMPLATE_REC"
  type="qipws:SEARCH_DNS_ACL_TEMPLATE_REC"/>

</xsd:schema>
```

DnsOptions.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Copyright (c) 2007, 2008 Alcatel-Lucent. -->

<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xmime="http://www.w3.org/2005/05/xmlmime"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  elementFormDefault="qualified">

  <xsd:annotation>
    <xsd:documentation>
      Each DNS option may take an argument.
      It may either be a single string, or an address match list.  Refer to
  xsd:choice.
      For the case of a single string, put it in the "value" element,
      for an address match list, use the "addrMatchValue" element.
      The exact syntax of the string argument depends on the option.
    </xsd:documentation>
  </xsd:annotation>

  <xsd:include schemaLocation="DnsAcl.xsd"/>

  <xsd:complexType name="ARRAY_OF_DNS_OPTIONS">
    <xsd:sequence>
      <!-- Use a <option/> element to specify an empty option list. -->
      <xsd:element name="option" type="qipws:DNS_OPTION_TYPE" minOccurs="0"
  maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="DNS_OPTION_TYPE">
    <xsd:sequence>
      <xsd:element name="name" type="xsd:string"/>
      <xsd:choice minOccurs="0">
        <xsd:element name="value" type="xsd:string"/>
        <xsd:element name="addrMatchValue" type="qipws:DNS_ACL_TYPE"/>
      </xsd:choice>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:simpleType name="DNS_OPTION_LEVEL_TYPE">
     <xsd:restriction base="xsd:string">
```

```
        <xsd:enumeration value="VIEW"/>
        <xsd:enumeration value="ZONE"/>
        <xsd:enumeration value="SERVER"/>
     </xsd:restriction>
  </xsd:simpleType>

</xsd:schema>
```

DnsSec.xsd

```
<?xml version="1.0"?>
<!-- Copyright (c) 2005-2008 Alcatel-Lucent. -->
<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xi="http://www.w3.org/2001/XInclude"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd"/>

  <!-- define type DNSSEC KEY -->
  <xsd:complexType name="DNSSEC_KEY_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="zoneName" type="xsd:string" minOccurs="0"/>
          <xsd:element name="keyName" type="xsd:string"/>
          <xsd:element name="keyTag" type="xsd:int"/>
          <xsd:element name="algorithm" type="qipws:DNSSEC_ALGORITHM_TYPE"
  minOccurs="0"/>
          <xsd:element name="bitSize" type="xsd:int" minOccurs="0"/>
          <xsd:element name="privateKey" type="xsd:string" minOccurs="0"/>
          <xsd:element name="publicKey" type="xsd:string" minOccurs="0"/>
          <xsd:element name="privateKeyFileName" type="xsd:string"
  minOccurs="0"/>
          <xsd:element name="publicKeyFileName" type="xsd:string"
  minOccurs="0"/>
          <xsd:element name="ksk" type="xsd:boolean" minOccurs="0"/>
          <xsd:element name="dsKey" type="xsd:string" minOccurs="0"/>
          <xsd:element name="deployDate" type="xsd:string" minOccurs="0"/>
          <xsd:element name="expireDate" type="xsd:string" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- Define Type DNSSEC_KEYS -->
  <xsd:complexType name="DNSSEC_KEY_KEYS">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="keyName" type="xsd:string"/>
```

```
            <xsd:element name="keyTag" type="xsd:int"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>

    <!-- define enum DNSSEC_ALGORITHM_TYPE -->
    <xsd:simpleType name="DNSSEC_ALGORITHM_TYPE">
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="1"/> <!-- RSA -->
        <xsd:enumeration value="1"/> <!-- RSAMD5-->
        <xsd:enumeration value="5"/> <!-- RSASHA1-->
      </xsd:restriction>
    </xsd:simpleType>

    <!-- define type ZONE -->
    <xsd:complexType name="ZONE_INFO_REC">
      <xsd:complexContent>
        <xsd:extension base="qipws:VQIP_BASE_ENTITY">
          <xsd:sequence>
            <xsd:element name="zoneName" type="xsd:string" minOccurs="0"/>
            <!--xsd:element name="parentZoneName" type="xsd:string"
    minOccurs="0"/-->
            <xsd:element name="childDomainNames" type="xsd:string"
    minOccurs="0" maxOccurs="unbounded"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:element name="DNSSEC_KEY_REC" type="qipws:DNSSEC_KEY_REC"/>
    <xsd:element name="ZONE_INFO_REC" type="qipws:ZONE_INFO_REC"/>

</xsd:schema>
```

5.7.DnsView.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Inserted into VQIPTypes.xsd via an XInclude. -->
<!-- Copyright (c) 2007, 2008 Alcatel-Lucent. -->

<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xmime="http://www.w3.org/2005/05/xmlmime"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
```

```
elementFormDefault="qualified">

<xsd:include schemaLocation="CoreTypes.xsd"/>
<xsd:include schemaLocation="DnsAcl.xsd"/>
<xsd:include schemaLocation="DnsOptions.xsd"/>

<!-- define DNS_VIEW_KEY -->
<xsd:complexType name="DNS_VIEW_KEY">
  <xsd:complexContent>
    <xsd:extension base="qipws:VQIP_KEY_ENTITY">
      <xsd:sequence>
        <xsd:element name="name" type="xsd:string"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!-- Define type DNS_VIEW_REC. -->
<xsd:complexType name="DNS_VIEW_REC">
  <xsd:complexContent>
    <xsd:extension base="qipws:VQIP_BASE_ENTITY">
      <xsd:sequence>
        <xsd:element name="name" type="xsd:string"/>
        <xsd:element name="order" type="xsd:int" minOccurs="0"/>
        <xsd:element name="matchClients" type="qipws:DNS_ACL_TYPE"
minOccurs="0"/>
        <xsd:element name="matchDestinations" type="qipws:DNS_ACL_TYPE"
minOccurs="0"/>
        <xsd:element name="matchRecursiveOnly" type="xsd:boolean"
minOccurs="0"/>
        <xsd:element name="dnsOptionList"
type="qipws:ARRAY_OF_DNS_OPTIONS" minOccurs="0"/>
        <xsd:element name="optionalAttributeList" type="qipws:UdaList"
minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!-- Define a SEARCH_DNS_VIEW_REC -->
<xsd:complexType name="SEARCH_DNS_VIEW_REC">
  <xsd:complexContent>
    <xsd:extension base="qipws:VQIP_BASE_ENTITY">
      <xsd:sequence>
        <xsd:element name="name" type="xsd:string" minOccurs="0"/>
```

```
            <xsd:element name="optionalAttributeList" type="qipws:UdaList"
   minOccurs="0"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>

   <!-- Schema for a DNS View's Option report. -->
   <xsd:element name="DnsViewOptionReport">
     <xsd:complexType>
       <xsd:sequence>
         <xsd:element name="option" type="qipws:DNS_VIEW_OPTIONRPT_OPT_TYPE"
   maxOccurs="unbounded" />
       </xsd:sequence>
     </xsd:complexType>
   </xsd:element>
   <xsd:complexType name="DNS_VIEW_OPTIONRPT_OPT_TYPE">
       <xsd:sequence>
         <xsd:element name="name" type="xsd:string"/>
         <xsd:element name="effectiveValue" type="xsd:string"/>
         <xsd:element name="viewValue" type="xsd:string" minOccurs="0"/>
         <xsd:element name="zoneValue" type="xsd:string" minOccurs="0"/>
         <xsd:element name="serverValue" type="xsd:string" minOccurs="0"/>
       </xsd:sequence>
   </xsd:complexType>


   <xsd:element name="DNS_VIEW_REC" type="qipws:DNS_VIEW_REC"/>
   <xsd:element name="DNS_VIEW_KEY" type="qipws:DNS_VIEW_KEY"/>
   <xsd:element name="SEARCH_DNS_VIEW_REC" type="qipws:SEARCH_DNS_VIEW_REC"/>

</xsd:schema>
5.8.DnsZone.xsd
<?xml version="1.0" encoding="UTF-8"?>
<!-- Inserted into VQIPTypes.xsd via an XInclude. -->
<!-- Copyright (c) 2007, 2008 Alcatel-Lucent. -->

<xsd:schema
   xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:xmime="http://www.w3.org/2005/05/xmlmime"
   targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
   elementFormDefault="qualified">

   <xsd:include schemaLocation="CoreTypes.xsd"/>
```

```
<xsd:include schemaLocation="DnsOptions.xsd"/>

<xsd:annotation>
  <xsd:documentation>
    Because zones can be global, specific to a view, shared, or a shared
reverse zone,
    uniquely specifing a zone requires not just its name, but also its
flavor:
    Two booleans in the zone's key determine if the zone is
    a shared v4 reverse zone (isSharedIntoView=true,
isIpv4ReverseZone=true),
    a shared forward zone (isSharedIntoView=true,
isIpv4ReverseZone=false), or
    view specific zone (isSharedIntoView=false, viewName has a value).
    Both booleans default to false if they are omitted.
    As of QIP 7.2, the zone's viewName must be populated.
  </xsd:documentation>
</xsd:annotation>

<xsd:complexType name="DNS_ZONE_KEY">
  <xsd:complexContent>
    <xsd:extension base="qipws:VQIP_KEY_ENTITY">
      <xsd:sequence>
        <xsd:element name="zoneName" type="xsd:string"/>
        <xsd:element name="viewName" type="xsd:string"/>
        <xsd:element name="sharedIntoView"  type="xsd:boolean"
minOccurs="0"/>
        <xsd:element name="ipv4ReverseZone" type="xsd:boolean"
minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="DNS_VIEW_ZONE_REC">
  <xsd:complexContent>
    <xsd:extension base="qipws:VQIP_BASE_ENTITY">
      <xsd:sequence>
        <xsd:element name="zone" type="qipws:DNS_ZONE_KEY"/>
        <xsd:element name="defaultTtl" type="xsd:long" minOccurs="0"/>
        <xsd:element name="dnsOptionList"
type="qipws:ARRAY_OF_DNS_OPTIONS" minOccurs="0"/>
        <xsd:element name="email" type="xsd:string" minOccurs="0"/>
        <xsd:element name="expireTime" type="xsd:long" minOccurs="0"/>
        <xsd:element name="negativeCacheTtl" type="xsd:long"
minOccurs="0"/>
```

```
          <xsd:element name="optionalAttributeList" type="qipws:UdaList"
minOccurs="0"/>
          <xsd:element name="postfixZoneExtension" type="xsd:string"
minOccurs="0"/>
          <xsd:element name="prefixZoneExtension" type="xsd:string"
minOccurs="0"/>
          <xsd:element name="refreshTime" type="xsd:long" minOccurs="0"/>
          <xsd:element name="retryTime" type="xsd:long" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="DNS_SHARED_ZONE_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="zone" type="qipws:DNS_ZONE_KEY"/>
          <!-- BEGIN: the following elements contain READ-ONLY data -->
            <xsd:element name="defaultTtl" type="xsd:long" minOccurs="0"/>
            <xsd:element name="email" type="xsd:string" minOccurs="0"/>
            <xsd:element name="expireTime" type="xsd:long" minOccurs="0"/>
            <xsd:element name="negativeCacheTtl" type="xsd:long"
minOccurs="0"/>
            <xsd:element name="optionalAttributeList" type="qipws:UdaList"
minOccurs="0"/>
            <xsd:element name="refreshTime" type="xsd:long" minOccurs="0"/>
            <xsd:element name="retryTime" type="xsd:long" minOccurs="0"/>
          <!-- END: READ-ONLY data elements. -->
          <xsd:element name="dnsOptionList"
type="qipws:ARRAY_OF_DNS_OPTIONS" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="SEARCH_DNS_VIEW_ZONE_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="zone" type="qipws:DNS_ZONE_KEY"/>
          <xsd:element name="optionalAttributeList" type="qipws:UdaList"
minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
```

```
  </xsd:complexType>

    <xsd:complexType name="SEARCH_DNS_SHARED_ZONE_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="zone" type="qipws:DNS_ZONE_KEY"/>
          <xsd:element name="optionalAttributeList" type="qipws:UdaList"
  minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- Declare the elements. -->
  <xsd:element name="DNS_ZONE_KEY"  type="qipws:DNS_ZONE_KEY"/>
  <xsd:element name="DNS_VIEW_ZONE_REC"  type="qipws:DNS_VIEW_ZONE_REC"/>
  <xsd:element name="DNS_SHARED_ZONE_REC"
  type="qipws:DNS_SHARED_ZONE_REC"/>
  <xsd:element name="SEARCH_DNS_VIEW_ZONE_REC"
  type="qipws:SEARCH_DNS_VIEW_ZONE_REC"/>
  <xsd:element name="SEARCH_DNS_SHARED_ZONE_REC"
  type="qipws:SEARCH_DNS_SHARED_ZONE_REC"/>

</xsd:schema>
```

5.9.DnsZoneServer.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Copyright (c) 2007, 2008 Alcatel-Lucent. -->

<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xmime="http://www.w3.org/2005/05/xmlmime"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd"/>
  <xsd:include schemaLocation="DnsOptions.xsd"/>
  <xsd:include schemaLocation="DnsZone.xsd"/>

  <xsd:complexType name="DNS_ZONE_SERVER_KEY">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string"/>
```

```
            <xsd:element name="zone" type="qipws:DNS_ZONE_KEY"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="DNS_ZONE_SERVER_REC">
      <xsd:complexContent>
        <xsd:extension base="qipws:VQIP_BASE_ENTITY">
          <xsd:sequence>
            <xsd:element name="name" type="xsd:string"/>
            <xsd:element name="zone" type="qipws:DNS_ZONE_KEY"/>
            <xsd:element name="dnsOptionList"
    type="qipws:ARRAY_OF_DNS_OPTIONS" minOccurs="0"/>
            <!-- Begin: Only for secondary servers -->
              <xsd:element name="primaryServer" type="xsd:string"
    minOccurs="0"/>
            <!-- End: Only secondary servers -->
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="SEARCH_DNS_ZONE_SERVER_REC">
      <xsd:complexContent>
        <xsd:extension base="qipws:VQIP_BASE_ENTITY">
          <xsd:sequence>
            <xsd:element name="zone" type="qipws:DNS_ZONE_KEY"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:element name="DNS_ZONE_SERVER_KEY" type="qipws:DNS_ZONE_SERVER_KEY"/>
    <xsd:element name="DNS_ZONE_SERVER_REC" type="qipws:DNS_ZONE_SERVER_REC"/>
    <xsd:element name="SEARCH_DNS_ZONE_SERVER_REC"
    type="qipws:SEARCH_DNS_ZONE_SERVER_REC"/>

</xsd:schema>

5.10.Info.xsd
<?xml version="1.0" encoding="UTF-8"?>
<!-- Inserted into VQIPTypes.xsd via an XInclude. -->
<!-- Copyright (c) 2007, 2008 Alcatel-Lucent. -->

<xsd:schema
```

```
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xmime="http://www.w3.org/2005/05/xmlmime"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd"/>

  <xsd:annotation>
    <xsd:documentation>
      Provides access to QIP information; these transaction(s) do not modify
   the database.
    </xsd:documentation>
  </xsd:annotation>

  <!-- Specify just the "name" when reading a file; results returned in
   "contents". -->
  <xsd:complexType name="INFO_CONFIGFILE_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string"/>
          <xsd:element name="contents" type="xsd:string" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="INFO_CONFIGFILE_REC" type="qipws:INFO_CONFIGFILE_REC"
   />

</xsd:schema>
```

5.11.IPv4ObjectAddr.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Inserted into VQIPTypes.xsd via an XInclude. -->
<!-- Copyright (c) 2007-2008 Alcatel-Lucent Technologies. -->

<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xmime="http://www.w3.org/2005/05/xmlmime"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd"/>
```

```
  <!-- Define IPV4_OBJECT_ADDR_KEY for API USE. -->
  <xsd:complexType name="IPV4_OBJECT_ADDR_KEY">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="ipAddrStr" type="xsd:string"/>
          <xsd:element name="fqdn" type="xsd:string"/>
          <xsd:element name="view" type="xsd:string" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- Define a primitive IPV4_OBJECT_ADDR_REC for qip-cli's FQDN handling. -
  ->
  <xsd:complexType name="IPV4_OBJECT_ADDR_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="ipAddrStr" type="xsd:string"/>
          <xsd:element name="fqdn" type="xsd:string"/>
          <xsd:element name="view" type="xsd:string" minOccurs="0"/>
          <xsd:element name="ttl" type="xsd:int" minOccurs="0"/>
          <xsd:element name="pubFwdZone" type="xsd:boolean" minOccurs="0"/>
          <xsd:element name="checkCollisions" type="xsd:boolean"
  minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="IPV4_OBJECT_ADDR_REC"
  type="qipws:IPV4_OBJECT_ADDR_REC"/>
  <xsd:element name="IPV4_OBJECT_ADDR_KEY"
  type="qipws:IPV4_OBJECT_ADDR_KEY"/>
</xsd:schema>
```

DnsView.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Inserted into VQIPTypes.xsd via an XInclude. -->
<!-- Copyright (c) 2007, 2008 Alcatel-Lucent. -->

<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xmime="http://www.w3.org/2005/05/xmlmime"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd"/>
  <xsd:include schemaLocation="DnsAcl.xsd"/>
  <xsd:include schemaLocation="DnsOptions.xsd"/>

  <!-- define DNS_VIEW_KEY -->
  <xsd:complexType name="DNS_VIEW_KEY">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- Define type DNS_VIEW_REC. -->
  <xsd:complexType name="DNS_VIEW_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string"/>
          <xsd:element name="order" type="xsd:int" minOccurs="0"/>
          <xsd:element name="matchClients" type="qipws:DNS_ACL_TYPE"
  minOccurs="0"/>
          <xsd:element name="matchDestinations" type="qipws:DNS_ACL_TYPE"
  minOccurs="0"/>
          <xsd:element name="matchRecursiveOnly" type="xsd:boolean"
  minOccurs="0"/>
          <xsd:element name="dnsOptionList"
  type="qipws:ARRAY_OF_DNS_OPTIONS" minOccurs="0"/>
          <xsd:element name="optionalAttributeList" type="qipws:UdaList"
  minOccurs="0"/>
```

```
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- Define a SEARCH_DNS_VIEW_REC -->
  <xsd:complexType name="SEARCH_DNS_VIEW_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string" minOccurs="0"/>
          <xsd:element name="optionalAttributeList" type="qipws:UdaList"
  minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- Schema for a DNS View's Option report. -->
  <xsd:element name="DnsViewOptionReport">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="option" type="qipws:DNS_VIEW_OPTIONRPT_OPT_TYPE"
  maxOccurs="unbounded" />
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:complexType name="DNS_VIEW_OPTIONRPT_OPT_TYPE">
      <xsd:sequence>
        <xsd:element name="name" type="xsd:string"/>
        <xsd:element name="effectiveValue" type="xsd:string"/>
        <xsd:element name="viewValue" type="xsd:string" minOccurs="0"/>
        <xsd:element name="zoneValue" type="xsd:string" minOccurs="0"/>
        <xsd:element name="serverValue" type="xsd:string" minOccurs="0"/>
      </xsd:sequence>
  </xsd:complexType>


  <xsd:element name="DNS_VIEW_REC" type="qipws:DNS_VIEW_REC"/>
  <xsd:element name="DNS_VIEW_KEY" type="qipws:DNS_VIEW_KEY"/>
  <xsd:element name="SEARCH_DNS_VIEW_REC" type="qipws:SEARCH_DNS_VIEW_REC"/>

</xsd:schema>
```

DnsZone.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Inserted into VQIPTypes.xsd via an XInclude. -->
<!-- Copyright (c) 2007, 20082007-2009 Alcatel-Lucent. -->

<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xmime="http://www.w3.org/2005/05/xmlmime"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd"/>
  <xsd:include schemaLocation="DnsOptions.xsd"/>

  <xsd:annotation>
    <xsd:documentation>
      Because zones can be global, specific to a view, shared, or a shared
  reverse zone,
      uniquely specifing a zone requires not just its name, but also its
  flavor:
      Two booleans in the zone's key determine if the zone is
      a shared v4 reverse zone (isSharedIntoView=true,
  isIpv4ReverseZone=true),
      a shared forward zone (isSharedIntoView=true,
  isIpv4ReverseZone=false), or
      view specific zone (isSharedIntoView=false, viewName has a value).
      Both booleans default to false if they are omitted.
      As of QIP 7.2, the zone's viewName must be populated.
    </xsd:documentation>
  </xsd:annotation>

  <xsd:complexType name="DNS_ZONE_KEY">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="zoneName" type="xsd:string"/>
          <xsd:element name="viewName" type="xsd:string" minOccurs=”0”/>
          <xsd:element name="sharedIntoView"  type="xsd:boolean"
  minOccurs="0"/>
          <xsd:element name="ipv4ReverseZone" type="xsd:boolean"
  minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
```

```
</xsd:complexType>

<xsd:complexType name="DNS_VIEW_ZONE_REC">
  <xsd:complexContent>
    <xsd:extension base="qipws:VQIP_BASE_ENTITY">
      <xsd:sequence>
        <xsd:element name="zone" type="qipws:DNS_ZONE_KEY"/>
        <xsd:element name="defaultTtl" type="xsd:long" minOccurs="0"/>
        <xsd:element name="dnsOptionList"
type="qipws:ARRAY_OF_DNS_OPTIONS" minOccurs="0"/>
        <xsd:element name="email" type="xsd:string" minOccurs="0"/>
        <xsd:element name="expireTime" type="xsd:long" minOccurs="0"/>
        <xsd:element name="negativeCacheTtl" type="xsd:long"
minOccurs="0"/>
        <xsd:element name="optionalAttributeList" type="qipws:UdaList"
minOccurs="0"/>
        <xsd:element name="postfixZoneExtension" type="xsd:string"
minOccurs="0"/>
        <xsd:element name="prefixZoneExtension" type="xsd:string"
minOccurs="0"/>
        <xsd:element name="refreshTime" type="xsd:long" minOccurs="0"/>
        <xsd:element name="retryTime" type="xsd:long" minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="DNS_SHARED_ZONE_REC">
  <xsd:complexContent>
    <xsd:extension base="qipws:VQIP_BASE_ENTITY">
      <xsd:sequence>
        <xsd:element name="zone" type="qipws:DNS_ZONE_KEY"/>
        <!-- BEGIN: the following elements contain READ-ONLY data -->
          <xsd:element name="defaultTtl" type="xsd:long" minOccurs="0"/>
          <xsd:element name="email" type="xsd:string" minOccurs="0"/>
          <xsd:element name="expireTime" type="xsd:long" minOccurs="0"/>
          <xsd:element name="negativeCacheTtl" type="xsd:long"
minOccurs="0"/>
          <xsd:element name="optionalAttributeList" type="qipws:UdaList"
minOccurs="0"/>
          <xsd:element name="refreshTime" type="xsd:long" minOccurs="0"/>
          <xsd:element name="retryTime" type="xsd:long" minOccurs="0"/>
        <!-- END: READ-ONLY data elements. -->
        <xsd:element name="dnsOptionList"
type="qipws:ARRAY_OF_DNS_OPTIONS" minOccurs="0"/>
      </xsd:sequence>
```

```
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="SEARCH_DNS_VIEW_ZONE_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="zone" type="qipws:DNS_ZONE_KEY"/>
          <xsd:element name="optionalAttributeList" type="qipws:UdaList"
  minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

    <xsd:complexType name="SEARCH_DNS_SHARED_ZONE_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="zone" type="qipws:DNS_ZONE_KEY"/>
          <xsd:element name="optionalAttributeList" type="qipws:UdaList"
  minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
<!-- When finding associations with zones, need to specify the type of
   association. -->
  <xsd:simpleType name="ZONE_INFRASTRUCTURE_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="NONMANAGEDSERVER"/>
    </xsd:restriction>
  </xsd:simpleType>

  <xsd:complexType name="ZONE_INFRASTRUCTURE_TYPE_KEY">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="qipws:ZONE_INFRASTRUCTURE_TYPE"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- Declare the elements. -->
```

```
   <xsd:element name="DNS_ZONE_KEY"  type="qipws:DNS_ZONE_KEY"/>
<xsd:element name="ZONE_INFRASTRUCTURE_TYPE_KEY"
   type="qipws:ZONE_INFRASTRUCTURE_TYPE_KEY"/>
   <xsd:element name="DNS_VIEW_ZONE_REC"  type="qipws:DNS_VIEW_ZONE_REC"/>
   <xsd:element name="DNS_SHARED_ZONE_REC"
   type="qipws:DNS_SHARED_ZONE_REC"/>
   <xsd:element name="SEARCH_DNS_VIEW_ZONE_REC"
   type="qipws:SEARCH_DNS_VIEW_ZONE_REC"/>
   <xsd:element name="SEARCH_DNS_SHARED_ZONE_REC"
   type="qipws:SEARCH_DNS_SHARED_ZONE_REC"/>

</xsd:schema>
```

DnsZoneServer.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Copyright (c) 2007, 2008 Alcatel-Lucent. -->

<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xmime="http://www.w3.org/2005/05/xmlmime"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd"/>
  <xsd:include schemaLocation="DnsOptions.xsd"/>
  <xsd:include schemaLocation="DnsZone.xsd"/>

  <xsd:complexType name="DNS_ZONE_SERVER_KEY">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string"/>
          <xsd:element name="zone" type="qipws:DNS_ZONE_KEY"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="DNS_ZONE_SERVER_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string"/>
          <xsd:element name="zone" type="qipws:DNS_ZONE_KEY"/>
          <xsd:element name="dnsOptionList"
  type="qipws:ARRAY_OF_DNS_OPTIONS" minOccurs="0"/>
          <!-- Begin: Only for secondary servers -->
            <xsd:element name="primaryServer" type="xsd:string"
  minOccurs="0"/>
          <!-- End: Only secondary servers -->
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="SEARCH_DNS_ZONE_SERVER_REC">
```

```
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="zone" type="qipws:DNS_ZONE_KEY"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="DNS_ZONE_SERVER_KEY" type="qipws:DNS_ZONE_SERVER_KEY"/>
  <xsd:element name="DNS_ZONE_SERVER_REC" type="qipws:DNS_ZONE_SERVER_REC"/>
  <xsd:element name="SEARCH_DNS_ZONE_SERVER_REC"
   type="qipws:SEARCH_DNS_ZONE_SERVER_REC"/>

</xsd:schema>
```

Info.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Inserted into VQIPTypes.xsd via an XInclude. -->
<!-- Copyright (c) 2007, 2008 Alcatel-Lucent. -->

<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xmime="http://www.w3.org/2005/05/xmlmime"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd"/>

  <xsd:annotation>
    <xsd:documentation>
      Provides access to QIP information; these transaction(s) do not modify
   the database.
    </xsd:documentation>
  </xsd:annotation>

  <!-- Specify just the "name" when reading a file; results returned in
   "contents". -->
  <xsd:complexType name="INFO_CONFIGFILE_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string"/>
          <xsd:element name="contents" type="xsd:string" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="INFO_CONFIGFILE_REC" type="qipws:INFO_CONFIGFILE_REC"
   />

</xsd:schema>
```

IPv4ObjectAddr.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Inserted into VQIPTypes.xsd via an XInclude. -->
<!-- Copyright (c) 2007-2008 Alcatel-Lucent Technologies. -->

<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xmime="http://www.w3.org/2005/05/xmlmime"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd"/>

  <!-- Define IPV4_OBJECT_ADDR_KEY for API USE. -->
  <xsd:complexType name="IPV4_OBJECT_ADDR_KEY">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="ipAddrStr" type="xsd:string"/>
          <xsd:element name="fqdn" type="xsd:string" minOccurs="0"/>
          <xsd:element name="view" type="xsd:string" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- Define a primitive IPV4_OBJECT_ADDR_REC for qip-cli's FQDN handling. -
  ->
  <xsd:complexType name="IPV4_OBJECT_ADDR_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="ipAddrStr" type="xsd:string"/>
          <xsd:element name="fqdn" type="xsd:string"/>
          <xsd:element name="view" type="xsd:string" minOccurs="0"/>
          <xsd:element name="ttl" type="xsd:int" minOccurs="0"/>
          <xsd:element name="pubFwdZone" type="xsd:boolean" minOccurs="0"/>
          <xsd:element name="checkCollisions" type="xsd:boolean"
  minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
```

```
   <xsd:element name="IPV4_OBJECT_ADDR_REC"
    type="qipws:IPV4_OBJECT_ADDR_REC"/>
   <xsd:element name="IPV4_OBJECT_ADDR_KEY"
    type="qipws:IPV4_OBJECT_ADDR_KEY"/>
</xsd:schema>
```

Maintainer.xsd

```
<?xml version="1.0"?>
<!-- Copyright (c) 2005-2008 Alcatel-Lucent. -->
<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xi="http://www.w3.org/2001/XInclude"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd" />

  <!-- define enum ENCRYPTION_TYPE -->
  <xsd:simpleType name="ENCRYPTION_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="NONE"/>
      <xsd:enumeration value="CRYPT-PW"/>
      <xsd:enumeration value="MD5-PW"/>
    </xsd:restriction>
  </xsd:simpleType>

  <!-- Define Type MAINTAINER_KEYS -->
  <xsd:complexType name="MAINTAINER_KEYS">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- Define Type MAINTAINER_REC -->
  <xsd:complexType name="MAINTAINER_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="registry" type="qipws:REGISTRY_TYPE"/>
          <xsd:element name="name" type="xsd:string" minOccurs="0"/>
          <xsd:element name="emailAddress" type="xsd:string"
  minOccurs="0"/>
          <xsd:element name="encryptionType" type="qipws:ENCRYPTION_TYPE"
  minOccurs="0"/>
          <xsd:element name="password" type="xsd:string" minOccurs="0"/>
```

```
            <xsd:element name="prefix" type="xsd:string" minOccurs="0"/>
            <xsd:element name="description" type="xsd:string" minOccurs="0"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:element name="MAINTAINER_REC" type="qipws:MAINTAINER_REC"/>

</xsd:schema>
```

5.13.ManagedFile.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Inserted into VQIPTypes.xsd via an XInclude. -->
<!-- Copyright (c) 2007, 2008 Alcatel-Lucent. -->

<xsd:schema
   xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:xmime="http://www.w3.org/2005/05/xmlmime"
   targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
   elementFormDefault="qualified">

   <xsd:include schemaLocation="CoreTypes.xsd"/>
   <xsd:import namespace="http://www.w3.org/2005/05/xmlmime"
    schemaLocation="xmime.xsd"/>


   <!-- Define MANAGED_FILE_KEY for API use. -->
   <xsd:complexType name="MANAGED_FILE_KEY">
     <xsd:complexContent>
       <xsd:extension base="qipws:VQIP_KEY_ENTITY">
         <xsd:sequence>
           <xsd:element name="name" type="xsd:string"/>
         </xsd:sequence>
       </xsd:extension>
     </xsd:complexContent>
   </xsd:complexType>

   <!-- Define MANAGED_FILE_SERVER_TYPE corresponding to the internal "group"
    names. -->
   <xsd:simpleType name="MANAGED_FILE_SERVER_TYPE">
     <xsd:restriction base="xsd:string">
       <xsd:enumeration value="DNS"/>
       <xsd:enumeration value="DHCP"/>
```

```
      </xsd:restriction>
  </xsd:simpleType>

  <!-- Define MANAGED_FILE_SERVER -->
  <xsd:complexType name="MANAGED_FILE_SERVER">
    <xsd:all>
      <xsd:element name="name" type="xsd:string"/>
      <xsd:element name="type" type="qipws:MANAGED_FILE_SERVER_TYPE"/>
    </xsd:all>
  </xsd:complexType>

  <!-- Define Type MANAGED_FILE_SERVER _KEY -->
  <xsd:complexType name="MANAGED_FILE_SERVER_KEY">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string"/>
          <xsd:element name="type" type="qipws:MANAGED_FILE_SERVER_TYPE"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- Define ARRAY_OF_MANAGED_FILE_SERVERS -->
  <xsd:complexType name="ARRAY_OF_MANAGED_FILE_SERVERS">
    <xsd:sequence>
      <xsd:element name="serverData" type="qipws:MANAGED_FILE_SERVER"
  maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>

  <!-- Define MANAGED_FILE_REC
       NOTE: When used with a "get", then an MTOM attachment contains the
  actual file.
            When used with a "search", the files are not returned; binaryData
  is null.
  -->
  <xsd:complexType name="MANAGED_FILE_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string"/>
          <xsd:element name="optionalAttributeList" type="qipws:UdaList"
  minOccurs="0"/>
          <xsd:element name="serverList"
  type="qipws:ARRAY_OF_MANAGED_FILE_SERVERS" minOccurs="0" />
```

```
            <xsd:element name="binaryData" type="xmime:base64Binary"
   minOccurs="0"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>

   <!-- Define type SEARCH_MANAGED_FILE_REC -->
   <xsd:complexType name="SEARCH_MANAGED_FILE_REC">
     <xsd:complexContent>
       <xsd:extension base="qipws:VQIP_BASE_ENTITY">
         <xsd:sequence>
           <xsd:element name="name" type="xsd:string" minOccurs="0"/>
           <xsd:element name="optionalAttributeList" type="qipws:UdaList"
   minOccurs="0"/>
           <xsd:element name="server" type="qipws:MANAGED_FILE_SERVER"
   minOccurs="0"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>

   <!-- Declare the elements. -->
   <xsd:element name="MANAGED_FILE_REC" type="qipws:MANAGED_FILE_REC"/>
   <xsd:element name="MANAGED_FILE_KEY" type="qipws:MANAGED_FILE_KEY"/>
   <xsd:element name="SEARCH_MANAGED_FILE_REC"
    type="qipws:SEARCH_MANAGED_FILE_REC"/>

</xsd:schema>
```

ManagedFile.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Inserted into VQIPTypes.xsd via an XInclude. -->
<!-- Copyright (c) 2007, 2008 Alcatel-Lucent. -->

<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xmime="http://www.w3.org/2005/05/xmlmime"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd"/>
  <xsd:import namespace="http://www.w3.org/2005/05/xmlmime"
   schemaLocation="xmime.xsd"/>


  <!-- Define MANAGED_FILE_KEY for API use. -->
  <xsd:complexType name="MANAGED_FILE_KEY">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- Define MANAGED_FILE_SERVER_TYPE corresponding to the internal "group"
  names. -->
  <xsd:simpleType name="MANAGED_FILE_SERVER_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="DNS"/>
      <xsd:enumeration value="DHCP"/>
    </xsd:restriction>
  </xsd:simpleType>

  <!-- Define MANAGED_FILE_SERVER -->
  <xsd:complexType name="MANAGED_FILE_SERVER">
    <xsd:sequence>
      <xsd:element name="name" type="xsd:string"/>
      <xsd:element name="type" type="qipws:MANAGED_FILE_SERVER_TYPE"/>
    </xsd:sequence>
  </xsd:complexType>
```

```
<!-- Define Type MANAGED_FILE_SERVER _KEY -->
<xsd:complexType name="MANAGED_FILE_SERVER_KEY">
  <xsd:complexContent>
    <xsd:extension base="qipws:VQIP_KEY_ENTITY">
      <xsd:sequence>
        <xsd:element name="name" type="xsd:string"/>
        <xsd:element name="type" type="qipws:MANAGED_FILE_SERVER_TYPE"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!-- Define ARRAY_OF_MANAGED_FILE_SERVERS -->
<xsd:complexType name="ARRAY_OF_MANAGED_FILE_SERVERS">
  <xsd:sequence>
    <xsd:element name="serverData" type="qipws:MANAGED_FILE_SERVER"
maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>

<!-- Define MANAGED_FILE_REC
     NOTE: When used with a "get", then an MTOM attachment contains the
actual file.
          When used with a "search", the files are not returned; binaryData
is null.
-->
<xsd:complexType name="MANAGED_FILE_REC">
  <xsd:complexContent>
    <xsd:extension base="qipws:VQIP_BASE_ENTITY">
      <xsd:sequence>
        <xsd:element name="name" type="xsd:string"/>
        <xsd:element name="optionalAttributeList" type="qipws:UdaList"
minOccurs="0"/>
        <xsd:element name="serverList"
type="qipws:ARRAY_OF_MANAGED_FILE_SERVERS" minOccurs="0" />
        <xsd:element name="binaryData" type="xmime:base64Binary"
minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!-- Define type SEARCH_MANAGED_FILE_REC -->
<xsd:complexType name="SEARCH_MANAGED_FILE_REC">
  <xsd:complexContent>
```

```
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string" minOccurs="0"/>
          <xsd:element name="optionalAttributeList" type="qipws:UdaList"
  minOccurs="0"/>
          <xsd:element name="server" type="qipws:MANAGED_FILE_SERVER"
  minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- Declare the elements. -->
  <xsd:element name="MANAGED_FILE_REC" type="qipws:MANAGED_FILE_REC"/>
  <xsd:element name="MANAGED_FILE_KEY" type="qipws:MANAGED_FILE_KEY"/>
  <xsd:element name="SEARCH_MANAGED_FILE_REC"
  type="qipws:SEARCH_MANAGED_FILE_REC"/>

</xsd:schema>
```

Node.xsd

```
<?xml version="1.0"?>
<!-- Copyright (c) 2005-2008 Alcatel-Lucent. -->
<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xi="http://www.w3.org/2001/XInclude"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd" />
  <xsd:include schemaLocation="Subnet.xsd" />

  <!--  define type DOMAIN_INFO_REC  -->
        <xsd:complexType name="DOMAIN_INFO_REC">
                <xsd:complexContent>
                        <xsd:extension base="qipws:VQIP_BASE_ENTITY">
                                <xsd:sequence>
                                        <xsd:element name="fqdn"
  type="xsd:string" minOccurs="0"/>
                                        <xsd:element name="viewName"
  type="xsd:string" minOccurs="0"/>
                                      <xsd:element name="TTL" type="xsd:int"
  minOccurs="0"/>
                                        <xsd:element name="pubFwdZone"
  type="xsd:boolean" minOccurs="0"/>
                                        <xsd:element name="pubRevZone"
  type="xsd:boolean" minOccurs="0"/>
                                        <xsd:element
  name="checkAllNameCollisions" type="xsd:boolean" minOccurs="0"/>
                                </xsd:sequence>
                        </xsd:extension>
                </xsd:complexContent>
        </xsd:complexType>

     <!-- Define Type DOMAIN_KEYS -->
        <xsd:complexType name="DOMAIN_KEYS">
                <xsd:complexContent>
                        <xsd:extension base="qipws:VQIP_KEY_ENTITY">
                                <xsd:sequence>
                                        <xsd:element name="fqdn"
  type="xsd:string"/>
                                        <xsd:element name="ipaddressKey"
  type="qipws:IPADDR_KEYS" minOccurs="0"/>
                                </xsd:sequence>
```

```
                    </xsd:extension>
                </xsd:complexContent>
            </xsd:complexType>


<!-- Define Type INTERFACE_KEYS -->
<xsd:complexType name="INTERFACE_KEYS">
  <xsd:complexContent>
    <xsd:extension base="qipws:VQIP_KEY_ENTITY">
      <xsd:sequence>
        <xsd:element name="name" type="xsd:string"/>
        <xsd:element name="nodeKey" type="qipws:NODE_KEYS"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!-- define struct INTERFACE_REC -->
<xsd:complexType name="INTERFACE_REC">
  <xsd:complexContent>
    <xsd:extension base="qipws:VQIP_BASE_ENTITY">
      <xsd:sequence>
        <xsd:element name="name" type="xsd:string" minOccurs="0"/>
        <xsd:element name="macAddress" type="xsd:string" minOccurs="0"/>
        <xsd:element name="ipAddrs" type="qipws:IPADDR_REC" minOccurs="0"
maxOccurs="unbounded"/>
        <xsd:element name="optionalAttributeList" type="qipws:UdaList"
minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!-- Define Type IPADDR_KEYS -->
<xsd:complexType name="IPADDR_KEYS">
  <xsd:complexContent>
    <xsd:extension base="qipws:VQIP_KEY_ENTITY">
      <xsd:sequence>
        <xsd:element name="ipAddrStr" type="xsd:string"/>
        <xsd:element name="interfaceKey" type="qipws:INTERFACE_KEYS"
minOccurs="0"/>
        <xsd:element name="subnetStrtAddr" type="qipws:V6_SUBNET_KEYS"
minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
```

```
</xsd:complexType>

<!-- IPADDR_REC -->
<xsd:complexType name="IPADDR_REC">
  <xsd:complexContent>
    <xsd:extension base="qipws:VQIP_BASE_ENTITY">
      <xsd:sequence>
        <xsd:element name="ipAddrStr" type="xsd:string" minOccurs="0"/>
        <xsd:element name="ipGroupName" type="xsd:string" minOccurs="0"/>
        <xsd:element name="v4ObjectName" type="xsd:string"
minOccurs="0"/>
        <xsd:element name="allocationAlgorithm"
type="qipws:IPADDR_ALLOC_ALG_TYPE" minOccurs="0"/>
        <xsd:element name="subnetStrtAddr" type="xsd:string"
minOccurs="0"/>
        <xsd:element name="domainInfo" type="qipws:DOMAIN_INFO_REC"
minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="optionalAttributeList" type="qipws:UdaList"
minOccurs="0"/>
        <xsd:element name="v4ObjectUdfList" type="qipws:UdfList"
minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

  <!-- define element MoveObjectRequest data -->
<xsd:element name="MoveObjectRequest">
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_REQUEST">
        <xsd:sequence>
         <xsd:element name="objKey" type="qipws:VQIP_KEY_ENTITY"/>
         <xsd:element name="toKey" type="qipws:VQIP_KEY_ENTITY"/>
    </xsd:sequence>
      </xsd:extension>
  </xsd:complexContent>
  </xsd:complexType>
</xsd:element>

<!-- define element LinkAddressToInterfaceRequest data -->
<xsd:element name="LinkAddressToInterfaceRequest">
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_REQUEST">
        <xsd:sequence>
```

```
            <xsd:element name="ipAddrStr" type="xsd:string"/>
            <xsd:element name="interfaceKey" type="qipws:INTERFACE_KEYS"/>
          </xsd:sequence>
        </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>

  <!-- define element UnlinkAddressFromInterfaceRequest data -->
       <xsd:element name="UnlinkAddressFromInterfaceRequest">
            <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="qipws:VQIP_BASE_REQUEST">
          <xsd:sequence>
            <xsd:element name="ipAddrStr" type="xsd:string"/>
            <xsd:element name="interfaceKey" type="qipws:INTERFACE_KEYS"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
               </xsd:complexType>
       </xsd:element>

  <!-- Define Type NODE_KEYS -->
  <xsd:complexType name="NODE_KEYS">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="uniqueID" type="xsd:string"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- define struct NODE_REC -->
  <xsd:complexType name="NODE_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string" minOccurs="0"/>
          <xsd:element name="uniqueID" type="xsd:string" minOccurs="0"/>
          <xsd:element name="description" type="xsd:string" minOccurs="0"/>
          <xsd:element name="nodeType" type="xsd:string" minOccurs="0"/>
          <xsd:element name="interfaces" type="qipws:INTERFACE_REC"
  minOccurs="0" maxOccurs="unbounded"/>
```

```
          <xsd:element name="checkAllNameCollisions" type="xsd:boolean"
minOccurs="0"/>
          <xsd:element name="optionalAttributeList" type="qipws:UdaList"
minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- define struct SEARCH_NODE_REC -->
       <xsd:complexType name="SEARCH_NODE_REC">
              <xsd:complexContent>
                     <xsd:extension base="qipws:VQIP_BASE_ENTITY">
                            <xsd:sequence>
                                   <xsd:element name="nodeName"
type="xsd:string" minOccurs="0"/>
                                   <xsd:element name="nodeType"
type="xsd:string" minOccurs="0"/>
                                   <xsd:element name="uniqueID"
type="xsd:string" minOccurs="0"/>
                                   <xsd:element name="description"
type="xsd:string" minOccurs="0"/>
                                   <xsd:element name="nodeUdaList"
type="qipws:UdaList" minOccurs="0"/>
                                   <xsd:element name="interfaceName"
type="xsd:string" minOccurs="0"/>
                                   <xsd:element name="macAddress"
type="xsd:string" minOccurs="0"/>
                                  <xsd:element name="interfaceUdaList"
type="qipws:UdaList" minOccurs="0"/>
                                   <xsd:element name="ipGroupName"
type="xsd:string" minOccurs="0"/>
          <xsd:element name="v4ObjectName" type="xsd:string"
minOccurs="0"/>
                                   <xsd:element name="ipAddress"
type="xsd:string" minOccurs="0"/>
                                   <xsd:element name="addressType"
type="qipws:V4V6_TYPE" minOccurs="0"/>
                                   <xsd:element name="fqdn"
type="xsd:string" minOccurs="0"/>
                                   <xsd:element name="viewName"
type="xsd:string" minOccurs="0"/>
                                  <xsd:element name="ipAddressUdaList"
type="qipws:UdaList" minOccurs="0"/>
                                   <xsd:element name="subnetStrtAddr"
type="xsd:string" minOccurs="0"/>
                                   <xsd:element name="v4ObjectUdf"
type="qipws:NameValue" minOccurs="0"/>
```

```
                                    </xsd:sequence>
                                </xsd:extension>
                            </xsd:complexContent>
                        </xsd:complexType>


    <xsd:element name="INTERFACE_KEYS" type="qipws:INTERFACE_KEYS"/>
    <xsd:element name="INTERFACE_REC" type="qipws:INTERFACE_REC"/>
    <xsd:element name="IPADDR_KEYS" type="qipws:IPADDR_KEYS"/>
    <xsd:element name="IPADDR_REC" type="qipws:IPADDR_REC"/>
    <xsd:element name="NODE_KEYS" type="qipws:NODE_KEYS"/>
    <xsd:element name="NODE_REC" type="qipws:NODE_REC"/>
    <xsd:element name="SEARCH_NODE_REC" type="qipws:SEARCH_NODE_REC"/>
</xsd:schema>
```

Organization.xsd

```
<?xml version="1.0"?>
<!-- Copyright (c) 2005-2008 Alcatel-Lucent. -->
<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xi="http://www.w3.org/2001/XInclude"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd" />

    <!-- Define Type ORG_REC -->
  <xsd:complexType name="ORG_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="organization" type="xsd:string"
  minOccurs="0"/>
          <xsd:element name="orgID" type="xsd:int" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="ORG_REC" type="qipws:ORG_REC"/>

</xsd:schema>
```

Pool.xsd

```
<?xml version="1.0"?>
<!-- Copyright (c) 2005-2008 Alcatel-Lucent. -->
<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xi="http://www.w3.org/2001/XInclude"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd" />

  <!-- define enum ALLOCATION_ALGORITHM_TYPE -->
  <xsd:simpleType name="ALLOCATION_ALGORITHM_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="BEST_FIT_FROM_END"/>
      <xsd:enumeration value="BEST_FIT_FROM_START"/>
      <xsd:enumeration value="SPARSE_ALLOCATION"/>
    </xsd:restriction>
  </xsd:simpleType>

  <!-- define enum FREE_PENDING_BLOCK_TYPE -->
  <xsd:simpleType name="FREE_PENDING_BLOCK_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="FREE_AFTER_DAYS"/>
      <xsd:enumeration value="FREE_AFTER_DAYS_AND_WHOIS"/>
      <xsd:enumeration value="FREE_AFTER_DAYS_OR_WHOIS"/>
      <xsd:enumeration value="FREE_AFTER_WHOIS"/>
    </xsd:restriction>
  </xsd:simpleType>

  <!--  define type CHILD_POOL_REC  -->
  <xsd:complexType name="CHILD_POOL_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:POOL_REC">
        <xsd:sequence>
          <xsd:element name="grandParentName" type="xsd:string"
  minOccurs="0"/>
          <xsd:element name="recurseRequest" type="xsd:boolean"
  minOccurs="0"/>
          <xsd:element name="recurseRequestNotify" type="xsd:boolean"
  minOccurs="0"/>
          <xsd:element name="v4ReverseZoneTemplate" type="xsd:string"
  minOccurs="0"/>
```

```
            <xsd:element name="v4Rule" type="xsd:string" minOccurs="0"/>
            <xsd:element name="v6Rule" type="xsd:string" minOccurs="0"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>

    <!-- Define Type POOL_KEYS -->
    <xsd:complexType name="POOL_KEYS">
      <xsd:complexContent>
        <xsd:extension base="qipws:VQIP_KEY_ENTITY">
          <xsd:sequence>
            <xsd:element name="name" type="xsd:string"/>
            <xsd:element name="parentName" type="xsd:string" minOccurs="0"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>

    <!-- Define type POOL_REC -->
    <xsd:complexType name="POOL_REC">
      <xsd:complexContent>
        <xsd:extension base="qipws:VQIP_BASE_ENTITY">
          <xsd:sequence>
            <xsd:element name="name" type="xsd:string"/>
            <xsd:element name="parentName" type="xsd:string" minOccurs="0"/>
            <xsd:element name="contact" type="xsd:string" minOccurs="0"/>
            <xsd:element name="childAllocationNotify" type="xsd:boolean"
    minOccurs="0"/>
            <!-- Attributes for IPv4 -->
            <xsd:element name="subnetOrg" type="xsd:string" minOccurs="0"/>
            <xsd:element name="v4Algorithm"
    type="qipws:ALLOCATION_ALGORITHM_TYPE" minOccurs="0"/>
            <xsd:element name="v4MinimumSparseSize" type="xsd:short"
    minOccurs="0"/>
            <xsd:element name="v4MaxAllocation" type="xsd:short"
    minOccurs="0"/>
            <xsd:element name="v4HDThreshold" type="xsd:short"
    minOccurs="0"/>
            <!-- Attributes for IPv6 -->
            <xsd:element name="v6Algorithm"
    type="qipws:ALLOCATION_ALGORITHM_TYPE" minOccurs="0"/>
            <xsd:element name="v6MinimumSparseSize" type="xsd:short"
    minOccurs="0"/>
            <xsd:element name="v6MaxAllocation" type="xsd:short"
    minOccurs="0"/>
```

```
          <xsd:element name="v6HDThreshold" type="xsd:short"
minOccurs="0"/>
          <!--  UDA/UDF -->
          <xsd:element name="optionalAttributeList" type="qipws:UdaList"
minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!--  define type SEARCH_POOL_REC -->
  <xsd:complexType name="SEARCH_POOL_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string" minOccurs="0"/>
          <xsd:element name="optionalAttributeList" type="qipws:UdaList"
minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!--  define type SEED_POOL_REC -->
  <xsd:complexType name="SEED_POOL_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:POOL_REC">
        <xsd:sequence>
          <xsd:element name="registry" type="qipws:REGISTRY_TYPE"
minOccurs="0"/>
          <xsd:element name="allowPendingStatus" type="xsd:boolean"
minOccurs="0"/>
          <xsd:element name="blockAddedNotify" type="xsd:boolean"
minOccurs="0"/>
          <xsd:element name="freePendingBlockWhen"
type="qipws:FREE_PENDING_BLOCK_TYPE" minOccurs="0"/>
          <xsd:element name="freePendingBlockDays" type="xsd:short"
minOccurs="0"/>
          <xsd:element name="v4ReverseZoneTemplate" type="xsd:string"
minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="CHILD_POOL_REC" type="qipws:CHILD_POOL_REC"/>
```

```
  <xsd:element name="POOL_REC" type="qipws:POOL_REC"/>
  <xsd:element name="SEED_POOL_REC" type="qipws:SEED_POOL_REC"/>
  <xsd:element name="SEARCH_POOL_REC" type="qipws:SEARCH_POOL_REC"/>

</xsd:schema>
```

Rule.xsd

```
<?xml version="1.0"?>
<!-- Copyright (c) 2005-2008 Alcatel-Lucent. -->
<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xi="http://www.w3.org/2001/XInclude"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd" />

  <!-- define enum RULE_LEVEL -->
  <xsd:simpleType name="RULE_LEVEL">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="NORMAL"/>
      <xsd:enumeration value="ADVANCED"/>
      <xsd:enumeration value="EXPERT"/>
    </xsd:restriction>
  </xsd:simpleType>

  <!-- define struct RULE_KEYS -->
  <xsd:complexType name="RULE_KEYS">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- define enum RULE_TYPE -->
  <xsd:simpleType name="RULE_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="USED"/>
      <xsd:enumeration value="FREE"/>
      <xsd:enumeration value="RESERVED"/>
      <xsd:enumeration value="SITE"/>
    </xsd:restriction>
  </xsd:simpleType>

  <!-- define enum RULE_COMPONENT_TYPE -->
  <xsd:simpleType name="RULE_COMPONENT_TYPE">
```

```
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="USED"/>
      <xsd:enumeration value="RESERVED"/>
    </xsd:restriction>
  </xsd:simpleType>

  <!-- define struct RULE_COMPONENT -->
  <xsd:complexType name="RULE_COMPONENT">
    <xsd:sequence>
      <xsd:element name="addrTemplate" type="xsd:string" minOccurs="0"/>
      <xsd:element name="ruleType" type="qipws:RULE_COMPONENT_TYPE"
minOccurs="0"/>
      <xsd:element name="subnetProfTemplate" type="xsd:string"
minOccurs="0"/>
      <xsd:element name="subnetBlockSize" type="xsd:short" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>

  <!-- define type ARRAY_OF_RULE_COMPONENT -->
  <xsd:complexType name="ARRAY_OF_RULE_COMPONENT">
    <xsd:sequence>
      <xsd:element name="ruleComponentData" type="qipws:RULE_COMPONENT"
minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>


  <!-- define struct RULE_REC -->
  <xsd:complexType name="RULE_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string" minOccurs="0"/>
          <xsd:element name="ruleLevel" type="qipws:RULE_LEVEL"
minOccurs="0"/>
          <xsd:element name="ruleType" type="qipws:RULE_TYPE"
minOccurs="0"/>
          <xsd:element name="addrTemplate" type="xsd:string"
minOccurs="0"/>
          <xsd:element name="subnetProfTemplate" type="xsd:string"
minOccurs="0"/>
          <xsd:element name="description" type="xsd:string" minOccurs="0"/>
          <xsd:element name="requestedSize" type="xsd:short"
minOccurs="0"/>
          <xsd:element name="v4V6Rule" type="qipws:V4V6_TYPE"
minOccurs="0"/>
```

```
            <xsd:element name="ruleComponentList"
   type="qipws:ARRAY_OF_RULE_COMPONENT" minOccurs="0"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>

   <xsd:element name="RULE_REC" type="qipws:RULE_REC"/>

</xsd:schema>
```

SchedJob.xsd

```
<?xml version="1.0"?>
<!-- Copyright (c) 2005-2008 Alcatel-Lucent. -->
<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xi="http://www.w3.org/2001/XInclude"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd" />

  <!-- define enum SCHEDULE_JOB_STATUS_TYPE -->
  <xsd:simpleType name="SCHEDULE_JOB_STATUS_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="ALL"/>
      <xsd:enumeration value="RUNNING"/>
      <xsd:enumeration value="PENDING"/>
      <xsd:enumeration value="DELAYED"/>
      <xsd:enumeration value="SCHEDULED"/>
      <xsd:enumeration value="COMPLETED"/>
      <xsd:enumeration value="FAILED"/>
      <xsd:enumeration value="CANCELLED"/>
    </xsd:restriction>
  </xsd:simpleType>

  <!-- define enum SCHEDULE_JOB_NAME_TYPE -->
  <xsd:simpleType name="SCHEDULE_JOB_NAME_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="ALL"/>
      <xsd:enumeration value="ADD_V6_SUBNETS"/>
      <xsd:enumeration value="PUSH_DNS_SERVER"/>
      <xsd:enumeration value="PUSH_DNS_UPDATE"/>
      <xsd:enumeration value="RENUMBER_V6_SUBNET"/>
      <xsd:enumeration value="SPLIT_V6_SUBNET"/>
      <xsd:enumeration value="RUN_REPORT"/>
    </xsd:restriction>
  </xsd:simpleType>


  <!-- define struct SCHEDULE_JOB_REC -->
  <xsd:complexType name="SCHEDULE_JOB_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
```

```
        <xsd:sequence>
          <xsd:element name="id" type="xsd:int" minOccurs="0"/>
          <xsd:element name="name" type="xsd:string" minOccurs="0"/>
          <xsd:element name="description" type="xsd:string" minOccurs="0"/>
          <xsd:element name="userName" type="xsd:string" minOccurs="0"/>
          <xsd:element name="status" type="qipws:SCHEDULE_JOB_STATUS_TYPE"
  minOccurs="0"/>
          <xsd:element name="startTime" type="xsd:string" minOccurs="0"/>
          <xsd:element name="scheduledTime" type="xsd:string"
  minOccurs="0"/>
          <xsd:element name="completeTime" type="xsd:string"
  minOccurs="0"/>
          <xsd:element name="server" type="xsd:string" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <!-- define struct SEARCH_SCHEDULE_JOB_REC -->
  <xsd:complexType name="SEARCH_SCHEDULE_JOB_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="userName" type="xsd:string" minOccurs="0"/>
          <xsd:element name="status" type="qipws:SCHEDULE_JOB_STATUS_TYPE"
  minOccurs="0"/>
          <xsd:element name="beginScheduledTime" type="xsd:string"
  minOccurs="0"/>
          <xsd:element name="endScheduledTime" type="xsd:string"
  minOccurs="0"/>
          <xsd:element name="beginCompletedTime" type="xsd:string"
  minOccurs="0"/>
          <xsd:element name="endCompletedTime" type="xsd:string"
  minOccurs="0"/>
          <xsd:element name="name" type="qipws:SCHEDULE_JOB_NAME_TYPE"
  minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="SEARCH_SCHEDULE_JOB_REC"
  type="qipws:SEARCH_SCHEDULE_JOB_REC"/>
  <xsd:element name="SCHEDULE_JOB_REC" type="qipws:SCHEDULE_JOB_REC"/>


</xsd:schema>
```

Server.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Copyright (c) 2005-2008 Alcatel-Lucent. -->
<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xi="http://www.w3.org/2001/XInclude"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd" />

  <!-- define SERVER_TYPE  -->
  <xsd:simpleType name="SERVER_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="DNS"/>
      <xsd:enumeration value="DHCP"/>
    </xsd:restriction>
  </xsd:simpleType>

  <!-- define struct SERVER_KEY -->
  <xsd:complexType name="SERVER_KEY">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="host" type="xsd:string"/>
          <xsd:element name="domain" type="xsd:string"/>
          <xsd:element name="serverType" type="qipws:SERVER_TYPE"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

</xsd:schema>
```

Subnet.xsd

```
<?xml version="1.0"?>
<!-- Copyright (c) 2005-2008 Alcatel-Lucent. -->
<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xi="http://www.w3.org/2001/XInclude"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd" />


  <!-- define enum SUBNET_ALLOCATION_TYPE -->
  <xsd:simpleType name="SUBNET_ALLOCATION_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="AUTOMATIC"/>
      <xsd:enumeration value="EXPLICIT"/>
    </xsd:restriction>
  </xsd:simpleType>
  <!-- define enum SUBNET_STATUS_TYPE -->
  <xsd:simpleType name="SUBNET_STATUS_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="UNUSED"/>
      <xsd:enumeration value="USED"/>
      <xsd:enumeration value="SCHED_MOVE"/>
      <xsd:enumeration value="PLANNED_USE"/>
    </xsd:restriction>
  </xsd:simpleType>
  <!-- define enum SUBNET_WARNING_TYPE -->
  <xsd:simpleType name="SUBNET_WARNING_TYPE">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="NONE"/>
      <xsd:enumeration value="EMAIL"/>
      <xsd:enumeration value="VISUAL"/>
      <xsd:enumeration value="BOTH"/>
    </xsd:restriction>
  </xsd:simpleType>

  <!-- define struct SEARCH_V4_SUBNET_REC -->
  <xsd:complexType name="SEARCH_V4_SUBNET_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
```

```
          <xsd:element name="name" type="xsd:string" minOccurs="0"/>
          <xsd:element name="ipAddress" type="xsd:string" minOccurs="0"/>
          <xsd:element name="subnetLength" type="xsd:short" minOccurs="0"/>
          <xsd:element name="domain" type="xsd:string" minOccurs="0"/>
          <xsd:element name="networkAddress" type="xsd:string"
minOccurs="0"/>
          <xsd:element name="networkName" type="xsd:string" minOccurs="0"/>
          <xsd:element name="subnetOrg" type="xsd:string" minOccurs="0"/>
          <xsd:element name="ospfArea" type="xsd:string" minOccurs="0"/>
                  <xsd:element name="optionalAttributeList"
type="qipws:UdaList" minOccurs="0"/>
          <xsd:element name="udf" type="qipws:NameValue" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- define struct SEARCH_V6_SUBNET_REC -->
  <xsd:complexType name="SEARCH_V6_SUBNET_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string" minOccurs="0"/>
          <xsd:element name="ipAddress" type="xsd:string" minOccurs="0"/>
          <xsd:element name="subnetLength" type="xsd:short" minOccurs="0"/>
                  <xsd:element name="optionalAttributeList"
type="qipws:UdaList" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- define struct V6_SUBNET_KEYS -->
  <xsd:complexType name="V6_SUBNET_KEYS">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="startAddress" type="xsd:string"/>
          <xsd:element name="prefixLength" type="xsd:short" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- define struct V4_SUBNET_REC -->
  <xsd:complexType name="V4_SUBNET_REC">
```

```
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string" minOccurs="0"/>
          <xsd:element name="address" type="xsd:string" minOccurs="0"/>
          <xsd:element name="mask" type="xsd:string" minOccurs="0"/>
          <xsd:element name="subnetLength" type="xsd:short" minOccurs="0"/>
          <xsd:element name="networkAddress" type="xsd:string"
minOccurs="0"/>
          <xsd:element name="location" type="qipws:LOCATION_REC"
minOccurs="0"/>
          <xsd:element name="contact" type="qipws:CONTACT_REC"
minOccurs="0"/>
          <xsd:element name="ospfArea" type="xsd:string" minOccurs="0"/>
          <xsd:element name="subnetOrg" type="xsd:string" minOccurs="0"/>
          <xsd:element name="application" type="xsd:string" minOccurs="0"/>
          <xsd:element name="domains" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
          <xsd:element name="dnsServers" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
          <xsd:element name="timeServers" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
          <xsd:element name="routers" type="qipws:ROUTER_REC" minOccurs="0"
maxOccurs="unbounded"/>
          <xsd:element name="description" type="xsd:string" minOccurs="0"/>
          <xsd:element name="dhcpServer" type="xsd:string" minOccurs="0"/>
          <xsd:element name="dhcpOptTempl" type="xsd:string"
minOccurs="0"/>
          <xsd:element name="dhcpPolTempl" type="xsd:string"
minOccurs="0"/>
          <xsd:element name="tftpServer" type="xsd:string" minOccurs="0"/>
          <xsd:element name="showUsed" type="xsd:boolean" minOccurs="0"/>
          <xsd:element name="checkUse" type="xsd:boolean" minOccurs="0"/>
          <xsd:element name="sharedNetwork" type="xsd:string"
minOccurs="0"/>
          <xsd:element name="hardwareType" type="qipws:HW_TYPE"
minOccurs="0"/>
          <xsd:element name="primaryInterface" type="xsd:boolean"
minOccurs="0"/>
          <xsd:element name="warning" type="qipws:SUBNET_WARNING_TYPE"
minOccurs="0"/>
          <xsd:element name="warningPercent" type="xsd:short"
minOccurs="0"/>
          <xsd:element name="allowDHCPClientsModifyDynamicObjectRRs"
type="qipws:MODIFY_RRS_TYPE" minOccurs="0"/>
          <xsd:element name="status" type="qipws:SUBNET_STATUS_TYPE"
minOccurs="0"/>
```

```
            <xsd:element name="optionalAttributeList" type="qipws:UdaList"
   minOccurs="0"/>
            <xsd:element name="udfList" type="qipws:UdfList" minOccurs="0"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>

   <!-- define struct V6_SUBNET_REC -->
   <xsd:complexType name="V6_SUBNET_REC">
     <xsd:complexContent>
       <xsd:extension base="qipws:VQIP_BASE_ENTITY">
         <xsd:sequence>
           <xsd:element name="name" type="xsd:string" minOccurs="0"/>
           <xsd:element name="startAddress" type="xsd:string"
   minOccurs="0"/>
           <xsd:element name="subnetLength" type="xsd:short" minOccurs="0"/>
           <xsd:element name="description" type="xsd:string" minOccurs="0"/>
           <xsd:element name="defaultDomain" type="xsd:string"
   minOccurs="0"/>
           <xsd:element name="numToCreate" type="xsd:short" minOccurs="0"/>
           <xsd:element name="allocationType"
   type="qipws:SUBNET_ALLOCATION_TYPE" minOccurs="0"/>
           <xsd:element name="startType" type="qipws:START_TYPE"
   minOccurs="0"/>
           <xsd:element name="allocationAlgorithm"
   type="qipws:IPADDR_ALLOC_ALG_TYPE" minOccurs="0"/>
           <xsd:element name="addressTemplateName" type="xsd:string"
   minOccurs="0"/>
           <xsd:element name="optionalAttributeList" type="qipws:UdaList"
   minOccurs="0"/>
         </xsd:sequence>
       </xsd:extension>
     </xsd:complexContent>
   </xsd:complexType>

   <xsd:element name="SEARCH_V4_SUBNET_REC"
   type="qipws:SEARCH_V4_SUBNET_REC"/>
   <xsd:element name="SEARCH_V6_SUBNET_REC"
   type="qipws:SEARCH_V6_SUBNET_REC"/>
   <xsd:element name="V4_SUBNET_REC" type="qipws:V4_SUBNET_REC"/>
   <xsd:element name="V6_SUBNET_REC" type="qipws:V6_SUBNET_REC"/>

</xsd:schema>
```

Template.xsd

```
<?xml version="1.0"?>
<!-- Copyright (c) 2005-2008 Alcatel-Lucent. -->
<xsd:schema
  xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
  xmlns:xi="http://www.w3.org/2001/XInclude"
  elementFormDefault="qualified">

  <xsd:include schemaLocation="CoreTypes.xsd" />

  <!-- define type ARRAYOF_V4MANAGED_RANGE_REC -->
  <xsd:complexType name="ARRAY_OF_V4MANAGED_RANGE_REC">
    <xsd:sequence>
      <xsd:element name="managedRangeData" type="qipws:V4MANAGED_RANGE_REC"
  minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>

  <!-- define type ARRAYOF_V6MANAGED_RANGE_REC -->
  <xsd:complexType name="ARRAY_OF_V6MANAGED_RANGE_REC">
    <xsd:sequence>
      <xsd:element name="managedRangeData" type="qipws:V6MANAGED_RANGE_REC"
  minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>

  <!-- Define Type ADDR_TEMPL_KEYS -->
  <xsd:complexType name="ADDR_TEMPL_KEYS">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- define type ADDR_TEMPL_REC -->
  <xsd:complexType name="ADDR_TEMPL_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
```

```
          <xsd:element name="name" type="xsd:string" minOccurs="0"/>
          <xsd:element name="description" type="xsd:string" minOccurs="0"/>
          <xsd:element name="v4ManagedRanges"
type="qipws:ARRAY_OF_V4MANAGED_RANGE_REC" minOccurs="0"/>
          <xsd:element name="v6ManagedRanges"
type="qipws:ARRAY_OF_V6MANAGED_RANGE_REC" minOccurs="0"/>
          <xsd:element name="v4V6Type" type="qipws:V4V6_TYPE"
minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!--  define type SERVER_PAIR_REC -->
  <xsd:complexType name="SERVER_PAIR_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="primaryDnsServer" type="xsd:string"/>
          <xsd:element name="secondaryDnsServer" type="xsd:string"
minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- define type ARRAY_OF_SERVER_PAIR_REC -->
  <xsd:complexType name="ARRAY_OF_SERVER_PAIR_REC">
    <xsd:sequence>
      <xsd:element name="serverPairData" type="qipws:SERVER_PAIR_REC"
minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>

  <!-- Define Type SUBNET_PROF_TEMPL_KEYS -->
  <xsd:complexType name="SUBNET_PROF_TEMPL_KEYS">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_KEY_ENTITY">
        <xsd:sequence>
          <xsd:element name="name" type="xsd:string"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!--  define type SUBNET_PROF_TEMPL_REC -->
```

```
<xsd:complexType name="SUBNET_PROF_TEMPL_REC">
  <xsd:complexContent>
    <xsd:extension base="qipws:VQIP_BASE_ENTITY">
      <xsd:sequence>
        <xsd:element name="altDnsServer" type="xsd:string"
minOccurs="0"/>
        <xsd:element name="name" type="xsd:string" minOccurs="0"/>
        <xsd:element name="description" type="xsd:string" minOccurs="0"/>
        <xsd:element name="dhcpServer" type="xsd:string" minOccurs="0"/>
        <xsd:element name="dhcpOptTempl" type="xsd:string"
minOccurs="0"/>
        <xsd:element name="domain" type="xsd:string" minOccurs="0"/>
        <xsd:element name="prefDnsServer" type="xsd:string"
minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!-- Define Type REV_ZONE_TEMPL_KEYS -->
<xsd:complexType name="REV_ZONE_TEMPL_KEYS">
  <xsd:complexContent>
    <xsd:extension base="qipws:VQIP_KEY_ENTITY">
      <xsd:sequence>
        <xsd:element name="name" type="xsd:string"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!-- Define type REV_ZONE_TEMPL_REC -->
<xsd:complexType name="REV_ZONE_TEMPL_REC">
  <xsd:complexContent>
    <xsd:extension base="qipws:VQIP_BASE_ENTITY">
      <xsd:sequence>
        <xsd:element name="name" type="xsd:string" minOccurs="0"/>
        <xsd:element name="description" type="xsd:string" minOccurs="0"/>
        <xsd:element name="serverPairs"
type="qipws:ARRAY_OF_SERVER_PAIR_REC" minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!-- define struct V4MANAGED_RANGE_REC -->
<xsd:complexType name="V4MANAGED_RANGE_REC">
```

```
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="startOffset" type="xsd:long" minOccurs="0"/>
          <xsd:element name="endOffset" type="xsd:long" minOccurs="0"/>
          <xsd:element name="numberOfObjects" type="xsd:long"
  minOccurs="0"/>
          <xsd:element name="addressType" type="qipws:ADDRESS_TYPE"
  minOccurs="0"/>
          <xsd:element name="leaseTime" type="xsd:long" minOccurs="0"/>
          <xsd:element name="objectClass" type="xsd:string" minOccurs="0"/>
          <xsd:element name="userClass" type="xsd:string" minOccurs="0"/>
          <xsd:element name="vendorClass" type="xsd:string" minOccurs="0"/>
          <xsd:element name="domain" type="xsd:string" minOccurs="0"/>
          <xsd:element name="dhcpServer" type="xsd:string" minOccurs="0"/>
          <xsd:element name="dhcpOptTempl" type="xsd:string"
  minOccurs="0"/>
          <xsd:element name="dhcpScopePolTempl" type="xsd:string"
  minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- define struct V6MANAGED_RANGE_REC -->
  <xsd:complexType name="V6MANAGED_RANGE_REC">
    <xsd:complexContent>
      <xsd:extension base="qipws:VQIP_BASE_ENTITY">
        <xsd:sequence>
          <xsd:element name="startOffset" type="xsd:long" minOccurs="0"/>
          <xsd:element name="endOffset" type="xsd:long" minOccurs="0"/>
          <xsd:element name="numberOfObjects" type="xsd:long"
  minOccurs="0"/>
          <xsd:element name="addressType" type="qipws:ADDRESS_TYPE"
  minOccurs="0"/>
          <xsd:element name="domain" type="xsd:string" minOccurs="0"/>
          <xsd:element name="intf" type="xsd:string" minOccurs="0"/>
          <xsd:element name="nodeType" type="xsd:string" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>


  <xsd:element name="ADDR_TEMPL_REC" type="qipws:ADDR_TEMPL_REC"/>
  <xsd:element name="ADDR_TEMPL_KEYS" type="qipws:ADDR_TEMPL_KEYS"/>
```

```
   <xsd:element name="SUBNET_PROF_TEMPL_REC"
   type="qipws:SUBNET_PROF_TEMPL_REC"/>
   <xsd:element name="SUBNET_PROF_TEMPL_KEYS"
   type="qipws:SUBNET_PROF_TEMPL_KEYS"/>
   <xsd:element name="REV_ZONE_TEMPL_REC" type="qipws:REV_ZONE_TEMPL_REC"/>
   <xsd:element name="REV_ZONE_TEMPL_KEYS" type="qipws:REV_ZONE_TEMPL_KEYS"/>
   <xsd:element name="V4MANAGED_RANGE_REC" type="qipws:V4MANAGED_RANGE_REC"/>
   <xsd:element name="V6MANAGED_RANGE_REC" type="qipws:V6MANAGED_RANGE_REC"/>

</xsd:schema>
```

UdaDefinition.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Copyright (c) 2008 Alcatel-Lucent. -->
<xsd:schemaxmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
                        xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                      targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
   elementFormDefault="qualified">
    <xsd:include schemaLocation="CoreTypes.xsd"/>

    <xsd:simpleType name="UDA_DEFINITION_ATTRIBUTE_TYPE">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="BOOLEAN"/>
            <xsd:enumeration value="IPV4_ADDRESS"/>
            <xsd:enumeration value="IPV6_ADDRESS"/>
            <xsd:enumeration value="MULTILINED_TEXT"/>
            <xsd:enumeration value="NUMERIC"/>
            <xsd:enumeration value="LIST"/>
            <xsd:enumeration value="TEXT"/>
        </xsd:restriction>
    </xsd:simpleType>

    <xsd:simpleType name="UDA_INFRASTRUCTURE_TYPE">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="AC_DEVICE"/>
            <xsd:enumeration value="AC_SUBSCRIBER"/>
            <xsd:enumeration value="BLOCK"/>
            <xsd:enumeration value="DC_SERVER"/>
            <xsd:enumeration value="DHCP_SERVER"/>
            <xsd:enumeration value="DNS_SERVER"/>
            <xsd:enumeration value="DNS_VIEW"/>
            <xsd:enumeration value="INTERFACE"/>
            <xsd:enumeration value="IPV4_ADDRESS"/>
            <xsd:enumeration value="IPV6_ADDRESS"/>
            <xsd:enumeration value="IPV4_SUBNET"/>
            <xsd:enumeration value="IPV6_SUBNET"/>
            <xsd:enumeration value="MANAGED_FILE"/>
            <xsd:enumeration value="MYVIEW"/>
            <xsd:enumeration value="NETWORK"/>
            <xsd:enumeration value="NODE"/>
            <xsd:enumeration value="ORGANIZATION"/>
            <xsd:enumeration value="POOL"/>
            <xsd:enumeration value="SUBNET_ORGANIZATION"/>
            <xsd:enumeration value="ZONE"/>
            <!-- organization is a global uda -->
```

```
        </xsd:restriction>
    </xsd:simpleType>

    <xsd:complexType name="UDA_INFRASTRUCTURE_TYPE_KEY">
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_KEY_ENTITY">
                <xsd:sequence>
                    <xsd:element name="name"
type="qipws:UDA_INFRASTRUCTURE_TYPE"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="UDA_DEFINITION_REC">
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_BASE_ENTITY">
                <xsd:sequence>
                    <xsd:element name="name" type="xsd:string"/>
                    <xsd:element name="type"
type="qipws:UDA_DEFINITION_ATTRIBUTE_TYPE" minOccurs="0"/>
                   <xsd:element name="dropDownList" type="qipws:StringList"
minOccurs="0"/>
                  <!-- Elements "initialValue" and "readOnly" are not valid
when "type" is "LIST" -->
                    <xsd:element name="initialValue" type="xsd:string"
minOccurs="0"/>
                    <xsd:element name="readOnly" type="xsd:boolean"
minOccurs="0"/>
                    <!-- Elements "minimum" and "maximum" are only for types
"NUMERIC" and "TEXT". -->
                  <xsd:element name="minimum" type="xsd:int" minOccurs="0"/>
                  <xsd:element name="maximum" type="xsd:int" minOccurs="0"/>
                    <xsd:element name="required" type="xsd:boolean"
minOccurs="0"/>
                    <xsd:element name="inheritValue" type="xsd:boolean"
minOccurs="0"/>
                    <xsd:element name="preEditCallout" type="xsd:string"
minOccurs="0"/>
                    <xsd:element name="postEditCallout" type="xsd:string"
minOccurs="0"/>
                    <xsd:element name="validationCallout" type="xsd:string"
minOccurs="0"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>
```

```
  <xsd:complexType name="UDA_DEFINITION_KEY">
      <xsd:complexContent>
          <xsd:extension base="qipws:VQIP_KEY_ENTITY">
              <xsd:sequence>
                  <xsd:element name="name" type="xsd:string"/>
              </xsd:sequence>
          </xsd:extension>
      </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="UDAGROUP_DEFINITION_REC">
      <xsd:complexContent>
          <xsd:extension base="qipws:VQIP_BASE_ENTITY">
              <xsd:sequence>
                 <xsd:element name="name" type="xsd:string" minOccurs="0"/>
                  <xsd:element name="udaList" type="qipws:StringList"
minOccurs="0"/>
              </xsd:sequence>
          </xsd:extension>
      </xsd:complexContent>
  </xsd:complexType>

  <!-- UDA_LIST_BASE_ENTITY solely exists to avoid a "Choice" name space
clash in LIST_BASE_ENTITY. -->
  <xsd:complexType name="UDA_LIST_BASE_ENTITY">
      <xsd:complexContent>
          <xsd:extension base="qipws:LIST_BASE_ENTITY"/>
      </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="UDA_DEFINITION_LIST_REC">
      <xsd:complexContent>
          <xsd:extension base="qipws:UDA_LIST_BASE_ENTITY">
              <xsd:choice minOccurs="0" maxOccurs="unbounded">
                  <xsd:element name="uda" type="xsd:string"/>
                  <xsd:element name="udaGroup" type="xsd:string"/>
              </xsd:choice>
          </xsd:extension>
      </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="UDAGROUP_DEFINITION_KEY">
      <xsd:complexContent>
          <xsd:extension base="qipws:VQIP_KEY_ENTITY">
```

```
                <xsd:sequence>
                    <xsd:element name="name" type="xsd:string"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>


    <xsd:complexType name="MYVIEW_KEY">
        <xsd:annotation>
            <xsd:documentation>This key is only supported for UDA set value
operations</xsd:documentation>
        </xsd:annotation>
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_KEY_ENTITY">
                <xsd:sequence>
                    <xsd:element name="name" type="xsd:string"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="NETWORK_KEY">
        <xsd:annotation>
            <xsd:documentation>This key is only supported for UDA set value
operations</xsd:documentation>
        </xsd:annotation>
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_KEY_ENTITY">
                <xsd:sequence>
                    <xsd:element name="address" type="xsd:string"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="ORG_KEY">
        <xsd:annotation>
            <xsd:documentation>This key is only supported for UDA set value
operations</xsd:documentation>
        </xsd:annotation>
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_KEY_ENTITY">
                <xsd:sequence>
                    <xsd:element name="name" type="xsd:string"/>
                </xsd:sequence>
```

```
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="SUBNET_ORG_KEY">
        <xsd:annotation>
            <xsd:documentation>This key is only supported for UDA set value
operations</xsd:documentation>
        </xsd:annotation>
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_KEY_ENTITY">
                <xsd:sequence>
                    <xsd:element name="name" type="xsd:string"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="V4_SUBNET_KEYS">
        <xsd:annotation>
            <xsd:documentation>This key is only supported for UDA set value
operations</xsd:documentation>
        </xsd:annotation>
        <xsd:complexContent>
            <xsd:extension base="qipws:VQIP_KEY_ENTITY">
                <xsd:sequence>
                    <xsd:element name="address" type="xsd:string"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>

    <!-- define element SetUdaRequest data -->
    <xsd:element name="SetUdaRequest">
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="qipws:VQIP_BASE_REQUEST">
                    <xsd:sequence>
                        <xsd:element name="infrastructure"
type="qipws:VQIP_KEY_ENTITY"/>
                        <xsd:element name="udaList" type="qipws:UdaList"/>
                    </xsd:sequence>
                </xsd:extension>
            </xsd:complexContent>
        </xsd:complexType>
    </xsd:element>
```

```
    <xsd:element name="UDA_DEFINITION_REC" type="qipws:UDA_DEFINITION_REC"/>
    <xsd:element name="UDA_DEFINITION_KEY" type="qipws:UDA_DEFINITION_KEY"/>
    <xsd:element name="UDAGROUP_DEFINITION_REC"
   type="qipws:UDAGROUP_DEFINITION_REC"/>
    <xsd:element name="UDAGROUP_DEFINITION_KEY"
   type="qipws:UDAGROUP_DEFINITION_KEY"/>
    <xsd:element name="UDA_DEFINITION_LIST_REC"
   type="qipws:UDA_DEFINITION_LIST_REC"/>
    <xsd:element name="UDA_INFRASTRUCTURE_TYPE_KEY"
   type="qipws:UDA_INFRASTRUCTURE_TYPE_KEY"/>

    <xsd:element name="MYVIEW_KEY" type="qipws:MYVIEW_KEY"/>
    <xsd:element name="NETWORK_KEY" type="qipws:NETWORK_KEY"/>
    <xsd:element name="ORG_KEY" type="qipws:ORG_KEY"/>
    <xsd:element name="SUBNET_ORG_KEY" type="qipws:SUBNET_ORG_KEY"/>
    <xsd:element name="V4_SUBNET_KEYS" type="qipws:V4_SUBNET_KEYS"/>

</xsd:schema>
```

VitalQIPTypes.xsd

```
<?xml version="1.0"?>
<!-- Copyright (c) 2005-2008 Alcatel-Lucent. -->
<xsd:schema xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:xi="http://www.w3.org/2001/XInclude"
   targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
   elementFormDefault="qualified">
   <xsd:include schemaLocation="AcDevice.xsd"/>
   <xsd:include schemaLocation="AcSubscriber.xsd"/>
   <xsd:include schemaLocation="AddressRange.xsd"/>
   <xsd:include schemaLocation="Block.xsd"/>
   <xsd:include schemaLocation="DnsAcl.xsd"/>
   <xsd:include schemaLocation="DnsSec.xsd"/>
   <xsd:include schemaLocation="DnsView.xsd"/>
   <xsd:include schemaLocation="DnsZone.xsd"/>
   <xsd:include schemaLocation="DnsZoneServer.xsd"/>
   <xsd:include schemaLocation="Info.xsd"/>
   <xsd:include schemaLocation="IPv4ObjectAddr.xsd"/>
   <xsd:include schemaLocation="Maintainer.xsd"/>
   <!-- Do not include ManagedFile.xsd because its use of
             xmime causes a problem with global vs. default namespace.  -->
   <xsd:include schemaLocation="Node.xsd"/>
   <xsd:include schemaLocation="Organization.xsd"/>
   <xsd:include schemaLocation="Pool.xsd"/>
   <xsd:include schemaLocation="Rule.xsd"/>
   <xsd:include schemaLocation="SchedJob.xsd"/>
   <xsd:include schemaLocation="Server.xsd"/>
   <xsd:include schemaLocation="Subnet.xsd"/>
   <xsd:include schemaLocation="Template.xsd"/>
   <xsd:include schemaLocation="UdaDefinition.xsd"/>


</xsd:schema>
```

7 VitalQIP Web Service WSDL

Overview

Purpose

This chapter provides a VitalQIP Web Service WSDL.

Contents

This chapter covers these topics.

Download VitalQIP WSDL and Schema from Web Server

The WSDL and the schema files can be downloaded from VitalQIP Web Server using following URLs:

xsd

http://*<webServerAddr>*:*<webServerPort>*/ws/services/VQIPWebService?xsd=*<xsdName>*

- ***<webServerAddr>*** is either the hostname or IP address of the VitalQIP web server.
- ***<webServerPort>*** is the TCP/IP port on which the VitalQIP web server is configured to listen. The default value is 80.
- ***<xsdName>*** is name of a schema file to retrieve.

Example

```
http://10.2.0.40/ws/services/VQIPWebService?xsd=Block.xsd
```

Note: If the VitalQIP web server is configured to use SSL, the URL must start with https: and not http:.

wsdl

http://*<webServerAddr>*:*<webServerPort>*/ws/services/VQIPWebService?wsdl

Example

```
http://10.2.0.40/ws/services/VQIPWebService?wsdl
```

The wsdl imports the standard schema file *oasis-200401-wss-wssecurity-secext-1.0.xsd*. Depending on the client SOAP toolkit, its compiler may either use its local copy or download from the Internet. If you need to prevent the compiler from using the Internet, files listed below need to be manually downloaded and the corresponding **schemaLocation** attributes in the wsdl and the schema files will need to be updated to reference your local copy.

- *oasis-200401-wss-wssecurity-secext-1.0.xsd*
- *oasis-200401-wss-wssecurity-utility-1.0.xsd*
- *xml.xsd*
- *xmldsig-core-schema.xsd*

VitalQIP Web Service WSDL

The following is the VitalQIP Web Service WSDL.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Copyright (c) 2005-2008 Alcatel-Lucent. -->

<wsdl:definitions targetNamespace="http://alcatel-lucent.com/qip/nb/ws"
   name="VQIPWebService.wsdl"
     xmlns:qipws="http://alcatel-lucent.com/qip/nb/ws"
     xmlns="http://schemas.xmlsoap.org/wsdl/"
     xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
     xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema"
     xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
   wssecurity-secext-1.0.xsd">

   <wsdl:types>
     <xsd:schema elementFormDefault="qualified">
       <xsd:import schemaLocation="VitalQIPTypes.xsd"
   namespace="http://alcatel-lucent.com/qip/nb/ws"/>
       <!-- Special handling for ManagedFile because its usage of xmime.xsd
   affects namespace. -->
       <xsd:import schemaLocation="ManagedFile.xsd"
   namespace="http://alcatel-lucent.com/qip/nb/ws"/>
       <!-- Username Token Profile support. -->
       <xsd:import
         schemaLocation="http://docs.oasis-open.org/wss/2004/01/oasis-
   200401-wss-wssecurity-secext-1.0.xsd"
         namespace="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
   wssecurity-secext-1.0.xsd"/>
     </xsd:schema>
   </wsdl:types>

   <message name="AddContainedObjectRequest">
     <part name="parameters" element="qipws:AddContainedObjectRequest"/>
   </message>
   <message name="AddRequest">
     <part name="parameters" element="qipws:AddRequest"/>
   </message>
   <message name="AddSeedBlockToSeedPoolRequest">
     <part name="parameters" element="qipws:AddSeedBlockToSeedPoolRequest"/>
   </message>
   <message name="AllocateBlockToPoolMessageRequest">
     <part name="parameters"
   element="qipws:AllocateBlockToPoolMessageRequest"/>
```

```
</message>
<message name="AssociateRequest">
  <part name="parameters" element="qipws:AssociateRequest"/>
</message>
<message name="AssociateListRequest">
  <part name="parameters" element="qipws:AssociateListRequest"/>
</message>
<message name="ApplFaultException">
  <part name="fault" element="qipws:ApplFault"/>
</message>
<message name="BlockAllocResponse">
  <part name="parameters" element="qipws:BlockAllocResponse"/>
</message>
<message name="CancelRequest">
  <part name="parameters" element="qipws:CancelRequest"/>
</message>
<message name="CopyRequest">
  <part name="parameters" element="qipws:CopyRequest"/>
</message>
<message name="CopyRuleRequest">
  <part name="parameters" element="qipws:CopyRuleRequest"/>
</message>
<message name="CountAssociationsRequest">
  <part name="parameters" element="qipws:CountAssociationsRequest"/>
</message>
  <message name="CountAssociationsResponse">
  <part name="parameters" element="qipws:CountAssociationsResponse"/>
</message>
<message name="DeleteContainedObjectRequest">
  <part name="parameters" element="qipws:DeleteContainedObjectRequest"/>
</message>
<message name="DeleteRequest">
  <part name="parameters" element="qipws:DeleteRequest"/>
</message>
<message name="DeleteForcedRequest">
  <part name="parameters" element="qipws:DeleteForcedRequest"/>
</message>
<message name="DeleteSeedBlockFromSeedPoolRequest">
  <part name="parameters"
element="qipws:DeleteSeedBlockFromSeedPoolRequest"/>
</message>
<message name="DissociateRequest">
  <part name="parameters" element="qipws:DissociateRequest"/>
</message>
<message name="DissociateListRequest">
```

```
    <part name="parameters" element="qipws:DissociateListRequest"/>
  </message>
  <message name="ExpandBlockRequest">
    <part name="parameters" element="qipws:ExpandBlockRequest"/>
  </message>
  <message name="ExplicitBlockAllocMessageRequest">
    <part name="parameters"
 element="qipws:ExplicitBlockAllocMessageRequest"/>
  </message>
  <message name="FindAssociationsRequest">
    <part name="parameters" element="qipws:FindAssociationsRequest"/>
  </message>
  <message name="FindAssociationsResponse">
    <part name="parameters" element="qipws:FindAssociationsResponse"/>
  </message>
  <message name="FreeBlockRequest">
    <part name="parameters" element="qipws:FreeBlockRequest"/>
  </message>
  <message name="FreePendingBlockRequest">
    <part name="parameters" element="qipws:FreePendingBlockRequest"/>
  </message>
  <message name="GetRequest">
    <part name="parameters" element="qipws:GetRequest"/>
  </message>
  <message name="GetAllRequest">
    <part name="parameters" element="qipws:GetAllRequest"/>
  </message>
  <message name="GetAllResponse">
    <part name="parameters" element="qipws:GetAllResponse"/>
  </message>
  <message name="GetAssociatedComponentsRequest">
    <part name="parameters" element="qipws:GetAssociatedComponentsRequest"/>
  </message>
  <message name="GetAssociatedComponentsResponse">
    <part name="parameters"
 element="qipws:GetAssociatedComponentsResponse"/>
  </message>
  <message name="GetContainedObjectsRequest">
    <part name="parameters" element="qipws:GetContainedObjectsRequest"/>
  </message>
  <message name="GetContainedObjectsResponse">
    <part name="parameters" element="qipws:GetContainedObjectsResponse"/>
  </message>
  <message name="GetFileRequest">
    <part name="parameters" element="qipws:GetFileRequest"/>
```

```
</message>
<message name="GetResponse">
  <part name="parameters" element="qipws:GetResponse"/>
</message>
<message name="MergeRequest">
  <part name="parameters" element="qipws:MergeRequest"/>
</message>
<message name="MoveRequest">
  <part name="parameters" element="qipws:MoveRequest"/>
</message>
<message name="MoveObjectRequest">
  <part name="parameters" element="qipws:MoveObjectRequest"/>
</message>
<message name="LinkAddressToInterfaceRequest">
  <part name="parameters" element="qipws:LinkAddressToInterfaceRequest"/>
</message>
<message name="PageSearchRequest">
  <part name="parameters" element="qipws:PageSearchRequest"/>
</message>
<message name="PageSearchResponse">
  <part name="parameters" element="qipws:PageSearchResponse"/>
</message>
<message name="QuickBlockAllocMessageRequest">
  <part name="parameters" element="qipws:QuickBlockAllocMessageRequest"/>
</message>
<message name="RenumberRequest">
  <part name="parameters" element="qipws:RenumberRequest"/>
</message>
<message name="RecoverBlockMessageRequest">
  <part name="parameters" element="qipws:RecoverBlockMessageRequest"/>
</message>
<message name="RecoverBlockResponse">
  <part name="parameters" element="qipws:RecoverBlockResponse"/>
</message>
<message name="Response">
  <part name="parameters" element="qipws:Response"/>
</message>
<message name="ReturnToParentBlockRequest">
  <part name="parameters" element="qipws:ReturnToParentBlockRequest"/>
</message>
<message name="SearchRequest">
  <part name="parameters" element="qipws:SearchRequest"/>
</message>
<message name="SearchResponse">
  <part name="parameters" element="qipws:SearchResponse"/>
```

```
</message>
<message name="SequenceRequest">
  <part name="parameters" element="qipws:SequenceRequest"/>
</message>
<message name="SequenceAssociatedComponentsRequest">
  <part name="parameters"
element="qipws:SequenceAssociatedComponentsRequest"/>
</message>
<message name="SetUdaRequest">
  <part name="parameters" element="qipws:SetUdaRequest"/>
</message>
<message name="SplitRequest">
  <part name="parameters" element="qipws:SplitRequest"/>
</message>
<message name="UnlinkAddressFromInterfaceRequest">
  <part name="parameters"
element="qipws:UnlinkAddressFromInterfaceRequest"/>
</message>
<message name="UpdateRequest">
  <part name="parameters" element="qipws:UpdateRequest"/>
</message>
<message name="Header">
  <part name="Security" element="wsse:Security"/>
</message>


<portType name="VQIPManagerPortType">

  <operation name="VQIPManager_AddContainedObjectRequest">
    <input message="qipws:AddContainedObjectRequest" />
    <output message="qipws:Response" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_AddRequest">
    <input message="qipws:AddRequest" />
    <output message="qipws:Response" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_AddSeedBlockToSeedPoolRequest">
    <input message="qipws:AddSeedBlockToSeedPoolRequest" />
    <output message="qipws:Response" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_AllocateBlockToPoolRequest">
    <input message="qipws:AllocateBlockToPoolMessageRequest" />
```

```
    <output message="qipws:BlockAllocResponse" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_AssociateRequest">
    <input message="qipws:AssociateRequest" />
    <output message="qipws:Response" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_AssociateListRequest">
    <input message="qipws:AssociateListRequest" />
    <output message="qipws:Response" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_CancelRequest">
    <input message="qipws:CancelRequest" />
    <output message="qipws:Response" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_CopyRequest">
    <input message="qipws:CopyRequest"/>
    <output message="qipws:Response"/>
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_CopyRuleRequest">
    <input message="qipws:CopyRuleRequest"/>
    <output message="qipws:Response"/>
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_CountAssociationsRequest">
    <input message="qipws:CountAssociationsRequest"/>
    <output message="qipws:CountAssociationsResponse"/>
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_DeleteContainedObjectRequest">
    <input message="qipws:DeleteContainedObjectRequest"/>
    <output message="qipws:Response"/>
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_DeleteRequest">
    <input message="qipws:DeleteRequest" />
    <output message="qipws:Response" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_DeleteForcedRequest">
    <input message="qipws:DeleteForcedRequest" />
```

```
    <output message="qipws:Response" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_DeleteSeedBlockFromSeedPoolRequest">
    <input message="qipws:DeleteSeedBlockFromSeedPoolRequest" />
    <output message="qipws:Response" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_DissociateRequest">
    <input message="qipws:DissociateRequest" />
    <output message="qipws:Response" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_DissociateListRequest">
    <input message="qipws:DissociateListRequest" />
    <output message="qipws:Response" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_ExpandBlockRequest">
    <input message="qipws:ExpandBlockRequest" />
    <output message="qipws:Response" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_ExplicitBlockAllocRequest">
    <input message="qipws:ExplicitBlockAllocMessageRequest" />
    <output message="qipws:BlockAllocResponse" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_FindAssociationsRequest">
    <input message="qipws:FindAssociationsRequest"/>
    <output message="qipws:FindAssociationsResponse"/>
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_FreeBlockRequest">
    <input message="qipws:FreeBlockRequest" />
    <output message="qipws:Response" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_FreePendingBlockRequest">
    <input message="qipws:FreePendingBlockRequest" />
    <output message="qipws:Response" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_GetRequest">
    <input message="qipws:GetRequest" />
```

```
    <output message="qipws:GetResponse" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_GetAllRequest">
    <input message="qipws:GetAllRequest" />
    <output message="qipws:GetAllResponse" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_GetAssociatedComponentsRequest">
    <input message="qipws:GetAssociatedComponentsRequest" />
    <output message="qipws:GetAssociatedComponentsResponse" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_GetFileRequest">
    <input message="qipws:GetFileRequest" />
    <output message="qipws:GetResponse" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_GetContainedObjectsRequest">
    <input message="qipws:GetContainedObjectsRequest"/>
    <output message="qipws:GetContainedObjectsResponse"/>
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_MergeRequest">
    <input message="qipws:MergeRequest"/>
    <output message="qipws:Response"/>
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_MoveObjectRequest">
    <input message="qipws:MoveObjectRequest"/>
    <output message="qipws:Response"/>
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_MoveRequest">
    <input message="qipws:MoveRequest"/>
    <output message="qipws:Response"/>
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_LinkAddressToInterfaceRequest">
    <input message="qipws:LinkAddressToInterfaceRequest"/>
    <output message="qipws:Response"/>
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_PageSearchRequest">
    <input message="qipws:PageSearchRequest"/>
```

```
    <output message="qipws:PageSearchResponse"/>
    <fault message="qipws:ApplFaultException" name="ApplException"/>
  </operation>
  <operation name="VQIPManager_QuickBlockAllocRequest">
    <input message="qipws:QuickBlockAllocMessageRequest" />
    <output message="qipws:BlockAllocResponse" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_RecoverBlockRequest">
    <input message="qipws:RecoverBlockMessageRequest"/>
    <output message="qipws:RecoverBlockResponse"/>
    <fault message="qipws:ApplFaultException" name="ApplException"/>
  </operation>
  <operation name="VQIPManager_RenumberRequest">
    <input message="qipws:RenumberRequest"/>
    <output message="qipws:Response"/>
    <fault message="qipws:ApplFaultException" name="ApplException"/>
  </operation>
  <operation name="VQIPManager_ReturnToParentBlockRequest">
    <input message="qipws:ReturnToParentBlockRequest" />
    <output message="qipws:Response" />
    <fault message="qipws:ApplFaultException" name="ApplException" />
  </operation>
  <operation name="VQIPManager_SearchRequest">
    <input message="qipws:SearchRequest"/>
    <output message="qipws:SearchResponse"/>
    <fault message="qipws:ApplFaultException" name="ApplException"/>
  </operation>
  <operation name="VQIPManager_SequenceRequest">
    <input message="qipws:SequenceRequest"/>
    <output message="qipws:Response"/>
    <fault message="qipws:ApplFaultException" name="ApplException"/>
  </operation>
  <operation name="VQIPManager_SequenceAssociatedComponentsRequest">
    <input message="qipws:SequenceAssociatedComponentsRequest"/>
    <output message="qipws:Response"/>
    <fault message="qipws:ApplFaultException" name="ApplException"/>
  </operation>
  <operation name="VQIPManager_SetUdaRequest">
    <input message="qipws:SetUdaRequest"/>
    <output message="qipws:Response"/>
    <fault message="qipws:ApplFaultException" name="ApplException"/>
  </operation>
  <operation name="VQIPManager_SplitRequest">
    <input message="qipws:SplitRequest"/>
```

```
      <output message="qipws:Response"/>
      <fault message="qipws:ApplFaultException" name="ApplException"/>
    </operation>
    <operation name="VQIPManager_UnlinkAddressFromInterfaceRequest">
      <input message="qipws:UnlinkAddressFromInterfaceRequest"/>
      <output message="qipws:Response"/>
      <fault message="qipws:ApplFaultException" name="ApplException" />
    </operation>
    <operation name="VQIPManager_UpdateRequest">
      <input message="qipws:UpdateRequest" />
      <output message="qipws:Response" />
      <fault message="qipws:ApplFaultException" name="ApplException" />
    </operation>
  </portType>

  <binding name="VQIPBinding" type="qipws:VQIPManagerPortType">
    <soap:binding style="document"
 transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="VQIPManager_AddContainedObjectRequest">
      <soap:operation soapAction="VQIPManager_AddContainedObjectRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>
    <operation name="VQIPManager_AddRequest">
      <soap:operation soapAction="VQIPManager_AddRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>
    <operation name="VQIPManager_AddSeedBlockToSeedPoolRequest">
```

```
      <soap:operation
  soapAction="VQIPManager_AddSeedBlockToSeedPoolRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_AllocateBlockToPoolRequest">
      <soap:operation soapAction="VQIPManager_AllocateBlockToPoolRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_AssociateRequest">
      <soap:operation soapAction="VQIPManager_AssociateRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>
    <operation name="VQIPManager_AssociateListRequest">
      <soap:operation soapAction="VQIPManager_AssociateListRequest"/>
      <input>
        <soap:body use="literal" />
```

```
      <soap:header use="literal" message="qipws:Header" part="Security"/>
    </input>
    <output>
      <soap:body use="literal" />
    </output>
    <fault name="ApplException">
      <soap:fault name="ApplException" use="literal" />
    </fault>
  </operation>
  <operation name="VQIPManager_CancelRequest">
    <soap:operation soapAction="VQIPManager_CancelRequest"/>
    <input>
      <soap:body use="literal" />
      <soap:header use="literal" message="qipws:Header" part="Security"/>
    </input>
    <output>
      <soap:body use="literal" />
    </output>
    <fault name="ApplException">
      <soap:fault name="ApplException" use="literal" />
    </fault>
  </operation>

  <operation name="VQIPManager_CopyRequest">
    <soap:operation soapAction="VQIPManager_CopyRequest"/>
    <input>
      <soap:body use="literal" />
      <soap:header use="literal" message="qipws:Header" part="Security"/>
    </input>
    <output>
      <soap:body use="literal" />
    </output>
    <fault name="ApplException">
      <soap:fault name="ApplException" use="literal" />
    </fault>
  </operation>

  <operation name="VQIPManager_CopyRuleRequest">
    <soap:operation soapAction="VQIPManager_CopyRuleRequest"/>
    <input>
      <soap:body use="literal" />
      <soap:header use="literal" message="qipws:Header" part="Security"/>
    </input>
    <output>
      <soap:body use="literal" />
```

```
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_CountAssociationsRequest">
      <soap:operation soapAction="VQIPManager_CountAssociationsRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_DeleteContainedObjectRequest">
      <soap:operation
  soapAction="VQIPManager_DeleteContainedObjectRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_DeleteRequest">
      <soap:operation soapAction="VQIPManager_DeleteRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
```

```
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_DeleteForcedRequest">
      <soap:operation soapAction="VQIPManager_DeleteForcedRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_DeleteSeedBlockFromSeedPoolRequest">
      <soap:operation
  soapAction="VQIPManager_DeleteSeedBlockFromSeedPoolRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>
    <operation name="VQIPManager_DissociateRequest">
      <soap:operation soapAction="VQIPManager_DissociateRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>
```

```
<operation name="VQIPManager_DissociateListRequest">
  <soap:operation soapAction="VQIPManager_DissociateListRequest"/>
  <input>
    <soap:body use="literal" />
    <soap:header use="literal" message="qipws:Header" part="Security"/>
  </input>
  <output>
    <soap:body use="literal" />
  </output>
  <fault name="ApplException">
    <soap:fault name="ApplException" use="literal" />
  </fault>
</operation>
<operation name="VQIPManager_ExpandBlockRequest">
  <soap:operation soapAction="VQIPManager_ExpandBlockRequest"/>
  <input>
    <soap:body use="literal" />
    <soap:header use="literal" message="qipws:Header" part="Security"/>
  </input>
  <output>
    <soap:body use="literal" />
  </output>
  <fault name="ApplException">
    <soap:fault name="ApplException" use="literal" />
  </fault>
</operation>

<operation name="VQIPManager_ExplicitBlockAllocRequest">
  <soap:operation soapAction="VQIPManager_ExplicitBlockAllocRequest"/>
  <input>
    <soap:body use="literal" />
    <soap:header use="literal" message="qipws:Header" part="Security"/>
  </input>
  <output>
    <soap:body use="literal" />
  </output>
  <fault name="ApplException">
    <soap:fault name="ApplException" use="literal" />
  </fault>
</operation>
<operation name="VQIPManager_FindAssociationsRequest">
  <soap:operation soapAction="VQIPManager_FindAssociationsRequest"/>
  <input>
    <soap:body use="literal" />
    <soap:header use="literal" message="qipws:Header" part="Security"/>
```

```
    </input>
    <output>
      <soap:body use="literal" />
    </output>
    <fault name="ApplException">
      <soap:fault name="ApplException" use="literal" />
    </fault>
  </operation>
  <operation name="VQIPManager_FreeBlockRequest">
    <soap:operation soapAction="VQIPManager_FreeBlockRequest"/>
    <input>
      <soap:body use="literal" />
      <soap:header use="literal" message="qipws:Header" part="Security"/>
    </input>
    <output>
      <soap:body use="literal" />
    </output>
    <fault name="ApplException">
      <soap:fault name="ApplException" use="literal" />
    </fault>
  </operation>

  <operation name="VQIPManager_FreePendingBlockRequest">
    <soap:operation soapAction="VQIPManager_FreePendingBlockRequest"/>
    <input>
      <soap:body use="literal" />
      <soap:header use="literal" message="qipws:Header" part="Security"/>
    </input>
    <output>
      <soap:body use="literal" />
    </output>
    <fault name="ApplException">
      <soap:fault name="ApplException" use="literal" />
    </fault>
  </operation>
  <operation name="VQIPManager_GetRequest">
    <soap:operation soapAction="VQIPManager_GetRequest"/>
    <input>
      <soap:body use="literal" />
      <soap:header use="literal" message="qipws:Header" part="Security"/>
    </input>
    <output>
      <soap:body use="literal" />
    </output>
    <fault name="ApplException">
```

```
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_GetAllRequest">
      <soap:operation soapAction="VQIPManager_GetAllRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_GetAssociatedComponentsRequest">
      <soap:operation
  soapAction="VQIPManager_GetAssociatedComponentsRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_GetContainedObjectsRequest">
      <soap:operation soapAction="VQIPManager_GetContainedObjectsRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
```

```
</operation>

<operation name="VQIPManager_GetFileRequest">
  <soap:operation soapAction="VQIPManager_GetFileRequest"/>
  <input>
    <soap:body use="literal" />
    <soap:header use="literal" message="qipws:Header" part="Security"/>
  </input>
  <output>
    <soap:body use="literal" />
  </output>
  <fault name="ApplException">
    <soap:fault name="ApplException" use="literal" />
  </fault>
</operation>

<operation name="VQIPManager_MergeRequest">
  <soap:operation soapAction="VQIPManager_MergeRequest"/>
  <input>
    <soap:body use="literal" />
    <soap:header use="literal" message="qipws:Header" part="Security"/>
  </input>
  <output>
    <soap:body use="literal" />
  </output>
  <fault name="ApplException">
    <soap:fault name="ApplException" use="literal" />
  </fault>
</operation>

<operation name="VQIPManager_MoveObjectRequest">
  <soap:operation soapAction="VQIPManager_MoveObjectRequest"/>
  <input>
    <soap:body use="literal" />
    <soap:header use="literal" message="qipws:Header" part="Security"/>
  </input>
  <output>
    <soap:body use="literal" />
  </output>
  <fault name="ApplException">
    <soap:fault name="ApplException" use="literal" />
  </fault>
</operation>
<operation name="VQIPManager_MoveRequest">
  <soap:operation soapAction="VQIPManager_MoveRequest"/>
```

```
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_LinkAddressToInterfaceRequest">
      <soap:operation
  soapAction="VQIPManager_LinkAddressToInterfaceRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_PageSearchRequest">
      <soap:operation soapAction="VQIPManager_PageSearchRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_QuickBlockAllocRequest">
      <soap:operation soapAction="VQIPManager_QuickBlockAllocRequest"/>
      <input>
        <soap:body use="literal" />
```

```
      <soap:header use="literal" message="qipws:Header" part="Security"/>
    </input>
    <output>
      <soap:body use="literal" />
    </output>
    <fault name="ApplException">
      <soap:fault name="ApplException" use="literal" />
    </fault>
  </operation>

  <operation name="VQIPManager_RecoverBlockRequest">
    <soap:operation soapAction="VQIPManager_RecoverBlockRequest"/>
    <input>
      <soap:body use="literal" />
      <soap:header use="literal" message="qipws:Header" part="Security"/>
    </input>
    <output>
      <soap:body use="literal" />
    </output>
    <fault name="ApplException">
      <soap:fault name="ApplException" use="literal" />
    </fault>
  </operation>

  <operation name="VQIPManager_RenumberRequest">
    <soap:operation soapAction="VQIPManager_RenumberRequest"/>
    <input>
      <soap:body use="literal" />
      <soap:header use="literal" message="qipws:Header" part="Security"/>
    </input>
    <output>
      <soap:body use="literal" />
    </output>
    <fault name="ApplException">
      <soap:fault name="ApplException" use="literal" />
    </fault>
  </operation>
  <operation name="VQIPManager_ReturnToParentBlockRequest">
    <soap:operation soapAction="VQIPManager_ReturnToParentBlockRequest"/>
    <input>
      <soap:body use="literal" />
      <soap:header use="literal" message="qipws:Header" part="Security"/>
    </input>
    <output>
      <soap:body use="literal" />
```

```
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_SearchRequest">
      <soap:operation soapAction="VQIPManager_SearchRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_SequenceRequest">
      <soap:operation soapAction="VQIPManager_SequenceRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_SequenceAssociatedComponentsRequest">
      <soap:operation
  soapAction="VQIPManager_SequenceAssociatedComponentsRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
```

```
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_SetUdaRequest">
      <soap:operation soapAction="VQIPManager_SetUdaRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_SplitRequest">
      <soap:operation soapAction="VQIPManager_SplitRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>

    <operation name="VQIPManager_UnlinkAddressFromInterfaceRequest">
      <soap:operation
  soapAction="VQIPManager_UnlinkAddressFromInterfaceRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
```

```
    </operation>

    <operation name="VQIPManager_UpdateRequest">
      <soap:operation soapAction="VQIPManager_UpdateRequest"/>
      <input>
        <soap:body use="literal" />
        <soap:header use="literal" message="qipws:Header" part="Security"/>
      </input>
      <output>
        <soap:body use="literal" />
      </output>
      <fault name="ApplException">
        <soap:fault name="ApplException" use="literal" />
      </fault>
    </operation>
  </binding>

  <service name="VQIPWebService">
    <port name="VQIPManagerPort" binding="qipws:VQIPBinding">
      <soap:address
 location="http://127.0.0.1:8080/ws/services/VQIPWebService"/>
    </port>
  </service>


</wsdl:definitions>
```