



Cisco VFrame Data Center Programmer's Guide

Release 1.1

June 30th, 2007

Americas Headquarters

Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
<http://www.cisco.com>
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

Text Part Number: OL-10401-01

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

CCVP, the Cisco logo, and the Cisco Square Bridge logo are trademarks of Cisco Systems, Inc.; Changing the Way We Work, Live, Play, and Learn is a service mark of Cisco Systems, Inc.; and Access Registrar, Aironet, BPX, Catalyst, CCDA, CCDP, CCIE, CCIP, CCNA, CCNP, CCSP, Cisco, the Cisco Certified Internetwork Expert logo, Cisco IOS, Cisco Press, Cisco Systems, Cisco Systems Capital, the Cisco Systems logo, Cisco Unity, Enterprise/Solver, EtherChannel, EtherFast, EtherSwitch, Fast Step, Follow Me Browsing, FormShare, GigaDrive, HomeLink, Internet Quotient, IOS, iPhone, IP/TV, iQ Expertise, the iQ logo, iQ Net Readiness Scorecard, iQuick Study, LightStream, Linksys, MeetingPlace, MGX, Networking Academy, Network Registrar, *Packet*, PIX, ProConnect, ScriptShare, SMARTnet, StackWise, The Fastest Way to Increase Your Internet Quotient, and TransPath are registered trademarks of Cisco Systems, Inc. and/or its affiliates in the United States and certain other countries.

All other trademarks mentioned in this document or Website are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (0705R)

Cisco VFrame Data Center Programmer's Guide, Release 1.1

© 2007, Cisco Systems, Inc. All rights reserved.



CONTENTS

Preface	i
Audience	i
Conventions	ii
Product Documentation	iii
Obtaining Documentation, Obtaining Support, and Security Guidelines	iv

CHAPTER 1

Introduction	1-1
WSDL and SOAP	1-1
SOAP Binding	1-1
Sample SOAP Traffic	1-3
Interoperability Considerations	1-5
WSDL Files Location	1-5
Security Mechanism	1-5
Session Management	1-6
Synchronous and Asynchronous Functionality	1-6
Error Handling	1-6

CHAPTER 2

API Functions	2-1
API Overview	2-2
AboutVFrame	2-2
Credentials Management	2-2
Resource Management	2-2
Service Template Management	2-3
Service Network Design	2-3
Service Operations	2-3
Job Management	2-4
Notifications Management	2-4
Data Objects	2-5
Managed Object Reference	2-5
Composite Data Objects	2-6
Faults Objects	2-6
VFFault	2-6
VFAuthorizationFault	2-6

- VfInvalidValueFault 2-7
- API Functions 2-8
 - Required Inputs 2-8
 - AboutVframe 2-8
 - getVersion 2-8
 - Device Credentials 2-9
 - getCredentials 2-10
 - addCredentials 2-10
 - saveCredentials 2-11
 - removeCredentials 2-11
 - Resources 2-12
 - getResourcePools 2-13
 - getResources 2-13
 - labelResources 2-14
 - maintainResources 2-14
 - manageResources 2-15
 - unManageResources 2-15
 - getResourcePoolDetails 2-16
 - saveResourcePools 2-16
 - getLogicalElementsForResource 2-17
 - Service Templates 2-17
 - getServiceTemplates 2-17
 - exportServiceTemplate 2-18
 - importServiceTemplate 2-18
 - Service Design 2-19
 - getServiceNetworkList 2-19
 - getServiceElements 2-20
 - Service Operations 2-21
 - startServiceNetwork 2-21
 - stopServiceNetwork 2-21
 - verifyServiceNetwork 2-21
 - createLogicalServer 2-22
 - imageServers 2-22
 - deleteLogicalServers 2-23
 - getResourceMappingForServiceElement 2-23
 - getServiceEvents 2-23
 - executeServiceElementOperation 2-24
 - Jobs 2-24
 - getJobRuns 2-26
 - Notifications 2-27

getTopics	2-28
registerReceiver	2-29
unRegisterReceiver	2-29
listReceivers	2-30
Notification Reply Information	2-30
Notification Details	2-30
Service Template State Change Notification	2-31
Service Network State Change Notification	2-31
Service Element State Change Notification	2-32
Fault Change Notification	2-32
Job Completed Notification	2-33
Server Inventory Finished Notification	2-33
Logical Server Created Notification	2-34
Logical Server Deleted Notification	2-34
Resource LifeCycle State Change Notification	2-35
Resource Managed State Change Notification	2-35
Resource Fault State Change Notification	2-36

CHAPTER 3

VFrame Data Center API SDK	3-1
Unpacking the API/SDK Files	3-1
VFrame Directory Contents	3-2
Sample Client Utilities	3-2
Online Programmer's Reference	3-4



Preface

This preface describes how to use Cisco VFrame Data Center Programmer's Guide.

Audience

This document is for the application developers with expertise in implementing third party software command calls through the use of API functions. Furthermore, the administrator should have expertise in data center management, including server administration, storage administration, network administration, network security.

Conventions

This document uses the following conventions:

Item	Convention
Commands and keywords.	boldface font
Variables for which you supply values.	<i>italic</i> font
Optional command keywords. You do not have to select any options.	[enclosed in brackets]
Required command keyword to be selected from a set of options. You must choose one option.	{options enclosed in braces separated by vertical bar}
Displayed session and system information.	screen font
Information you enter.	boldface screen font
Variables you enter.	<i>italic screen</i> font
Menu items and button names.	boldface font
Selecting a menu item.	Option > Network Preferences

Notes use the following conventions:



Note

Means *reader take note*. Notes contain helpful suggestions or references to material not covered in the publication.

Cautions use the following conventions:



Caution

Means *reader be careful*. In this situation, you might do something that could result in equipment damage or loss of data.

Product Documentation

Table 1 describes the product documentation that is available. For information on ordering printed documents, see [Obtaining Documentation, Obtaining Support, and Security Guidelines, page iv](#).

Table 1 Product Documentation

Document Title	Available Formats
<i>Release Notes for Cisco VFrame Data Center 1.1</i>	On Cisco.com.
<i>Cisco VFrame Data Center 1.1 Installation and Configuration Guide</i>	<ul style="list-style-type: none"> On the product's recovery CD-ROM. On Cisco.com. Additional printed document available by order (part number DOC-7817674=).
<i>Cisco VFrame Data Center 1.1 Administration Guide</i>	<ul style="list-style-type: none"> On Cisco.com. Included in the online help in HTML and PDF formats.
<i>Cisco VFrame Data Center 1.1 Regulatory Compliance Information</i>	Printed document that was included with the product.
<i>Cisco VFrame Data Center 1.1 Programmer's Guide</i>	<ul style="list-style-type: none"> On Cisco.com.
<i>Cisco VFrame Data Center 1.1 Online Programmer's Reference</i>	<ul style="list-style-type: none"> On the product's recovery CD-ROM. By entering the VFrame server's IP address in a browser
<i>Important Safety Information</i>	Printed document that was included with the product.
Context-sensitive online help	<ul style="list-style-type: none"> Select Help > Contents to open the help system. Select Help > For This Page to obtain help on the page currently in the window. Click the Help button in a dialog box.

Obtaining Documentation, Obtaining Support, and Security Guidelines

For information on obtaining documentation, obtaining support, providing documentation feedback, security guidelines, and also recommended aliases and general Cisco documents, see the monthly *What's New* in Cisco Product Documentation, which also lists all new and revised Cisco technical documentation, at:

<http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html>



CHAPTER 1

Introduction

This document describes the VFrame API for the Cisco VFrame Data Center product. This document is intended for Cisco partners and customers who wish to integrate their products with VFrame Data Center. This document assumes the reader is familiar with the VFrame Data Center product and has a working knowledge of XML, Web Services, and SOAP.

The VFrame API exposes a subset of the VFrame Data Center functionality through a collection of SOAP based web services. Web services provide a set of standards based, platform independent protocols for communication between applications. Applications written in any programming language, for any platform, can integrate with VFrame Data Center through the VFrame API. This integration allows third party programmers access to VFrame operations.

Note that the VFrame API is not intended to provide all the functionality of the VFrame Data Center GUI. The VFrame API focuses on providing third-party management applications the ability to programmatically manage and monitor services offered by VFrame Data Center. The user is expected to use the VFrame Data Center GUI to do basic configuration such as configuring users, template design, device discovery, resource pool creation, and logical service design. The user can then make use of the VFrame API to perform operations such as managing device credentials, retrieving resource inventory, managing and un-managing resources, service operations, and event notifications.

WSDL and SOAP

WSDL stands for Web Services Description Language. WSDL is a document written in XML. The document describes a web service. It specifies the location of the service and the operations (or methods) the service exposes.

SOAP is a lightweight protocol for exchanging structured information in a decentralized, distributed environment. It is an XML based protocol that consists of three parts: an envelope that defines a framework for describing what is in a message and how to process it, a set of encoding rules for expressing instances of application-defined DataTypes, and a convention for representing remote procedure calls and responses.

For more information on SOAP 1.1, see the specification located at <http://www.w3.org/TR/soap/>.

SOAP Binding

A WSDL binding describes how the service is bound to a messaging protocol, particularly the SOAP messaging protocol. A WSDL SOAP binding can be either a Remote Procedure Call (RPC) style binding or a document style binding. A SOAP binding also can have an encoded use or a literal use.

VFrame API SOAP binding information is specified in the binding section of the Serviceability SOAP WSDL files. Binding specifications apply to both request and response messages.

SOAP Binding covers the aspects explained in the following sections.

Character Encoding

VFrame APIs use UTF-8 to encode the data stream in both request and response SOAP messages. The encoding attribute of the XML declaration specifies UTF-8 encoding. VFrame APIs also set “text/xml; charset=utf-8” as the value of the Content-Type response header field.

Binding Style

VFrame Web Services based API's follow the “document/literal wrapped” style WSDL to allow full parameter validation. Document/literal is WS-I compliant, and the wrapped pattern meets the WS-I restriction that the SOAP message's soap body has only one child.

Transport Protocols

SOAP allows different transport protocols to carry SOAP messages. VFrame APIs use the standard HTTPS as its transport. Clients use the POST method to send requests to VFrame APIs.

Encoding Rule

VFrame APIs follow the recommended data model and serialization/encoding rules, as defined in Section 5.1 of SOAP Specification 1.1, for both request and response messages. SOAP simple types are based on the built-in data types that are defined in the XML Schema, Part 2. VFrame APIs define their own data types, which are derived from the built-in types. The schemas element of the VFrame APIs WSDL file specifies the derived data types.

SOAPAction HTTP Header

The SOAP 1.1 specification says this about the HTTP SOAPAction header:

“The SOAPAction HTTP request header field can be used to indicate the intent of the SOAP HTTP request. The value is a URI identifying the intent. SOAP places no restrictions on the format or specificity of the URI or that it is resolvable. An HTTP client MUST use this header field when issuing a SOAP HTTP Request.”

VFrame APIs require SOAP clients to include the SOAP Action HTTP header field in the request message. For VFrame APIs, the soapAction attribute of the SOAP element, which is defined under the binding section of the WSDL files, is set to empty string (“”). The header field value of empty string (“”) means that the intent of the SOAP message is provided by the HTTP Request-URI

SOAP Header

The SOAP header provides a general way to add features to SOAP messages in a decentralized fashion with no prior contract between the sender and recipient. VFrame APIs do not use this feature, so no Header element is expected in the envelope, and is ignored.

VFrame Data Center uses Apache Axis as the SOAP engine.

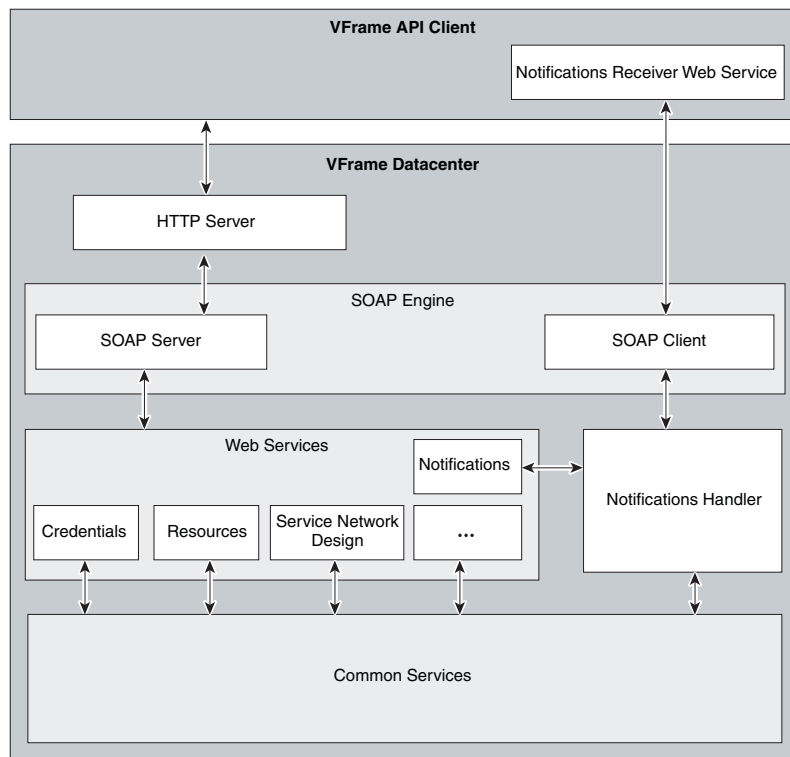


Note

Apache Axis is an open source, XML based Web service framework. It consists of a Java and a C++ implementation of the SOAP server, and various utilities and APIs for generating and deploying Web service applications. You can learn more about the Axis Web services framework at the Apache Software Foundation, <http://www.apache.org>.

Figure 1-1 shows an architectural overview of the VFrame API.

Figure 1-1 VFrame API Structural Diagram



Sample SOAP Traffic

The following section shows examples of SOAP request and response traffic.

SOAP Request

The following sample shows a typical SOAP request sent to an instance of the VFrame Data Center Server. In the SOAP message the `getServiceElements` operation is requested.

```
POST /services/ServiceDesignService HTTP/1.0
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.3
Host: vcc-server:8091
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""
```

```

Content-Length: 503
Authorization: Basic YWRtaW5AQWRtaW5Db250ZXh0OmNpc2NvMTIz
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <getServiceElements xmlns="urn:servicedesign.vfdc.cisco.com">
      <serviceElementQuery xmlns="">
        <serviceRef>
          <managedObjectId>
            <type>ServiceNetwork</type>
            <value>123345</value>
          </managedObjectId>
        </serviceRef>
      </serviceElementQuery>
    </getServiceElements>
  </soapenv:Body>
</soapenv:Envelope>

```

SOAP Response

The following sample shows a typical SOAP response sent from an instance of the VFrame Data Center Server. The SOAP message is based on the getVersion operation. The results of the operation are wrapped in the <getVersionResponse> element in the SOAP body.

```

HTTP/1.1 200 OK
Date: Thu, 14 Jun 2007 18:10:21 GMT
Server: Jetty/5.1.1 (Linux/2.4.21-47.7.ELsmp i386 java/1.5.0_11
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Set-Cookie: JSESSIONID=blgsf9q9bib2e;path=/
Content-Type: text/xml; charset=utf-8
Connection: close
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope
    xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getVersionResponse xmlns="urn:aboutvframe.vfdc.cisco.com">
        <versionInfo xmlns="">
          <version>1.1</version>
          <product>VFrame Data Center</product>
          <build>Sun May 27 10:02:23 PDT 2007</build>
        </versionInfo>
      </getVersionResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```

SOAP Fault Response

Responses to SOAP requests can take one of two forms - a success response or an error response. For an error response, the response could contain either HTTP errors or SOAP faults. When SOAP faults are generated, they are returned as HTTP 500 errors.

When a VFrame Data Center API processes a request and detects that an error occurred, it replies with a SOAP fault element in the response. The fault element appears as the first response body entry.

The following is an example of a SOAP fault response:

```
HTTP/1.1 500 Internal Server Error
Date: Thu, 14 Jun 2007 19:16:04 GMT
Server: Jetty/5.1.1 (Linux/2.4.21-47.7.ELsmp i386 java/1.5.0_11
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Set-Cookie: JSESSIONID=2cso11emc9ddl;path=/
Content-Type: text/xml; charset=utf-8
Connection: close
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <soapenv:Fault>
      <faultcode>soapenv:Server.generalException</faultcode>
      <faultstring></faultstring>
      <detail>
        <ns1:VFInvalidValueFault xmlns:ns1="urn:types.vfdc.cisco.com">
          <faultKey>INPUT_VALIDATION_FAIL</faultKey>
          <faultMessage>Invalid service network ref.</faultMessage>
        </ns1:VFInvalidValueFault>
        <ns2:exceptionName xmlns:ns2="http://xml.apache.org/axis/">
          com.cisco.nbv.sv.nbapi.types.VFInvalidValueFaultType
        </ns2:exceptionName>
        <ns3:hostname xmlns:ns3="http://xml.apache.org/axis/">
          vcc-29.nbv.cisco.com
        </ns3:hostname>
      </detail>
    </soapenv:Fault>
  </soapenv:Body>
</soapenv:Envelope>
```

Interoperability Considerations

All the Web Service APIs provided meet the WS-I Basic Profile interoperability requirements. The WS-I Basic Profile is a specification from the Web Services Interoperability Industry Consortium that provides interoperability guidance for core Web Services specifications such as SOAP, WSDL, and UDDI. The profile uses Web Services Description Language (WSDL) to enable the description of services as sets of endpoints operating on messages.

WSDL Files Location

The WSDL files are accessed from within the SDK, which is on the VFrame Data Center Appliance at the URL `http://<vframe-host>/sdk`, where `<vframe-host>` is the IP address (or name if your network supports DNS) of the VFrame Data Center Server. Here, there are two active links: one for the Cisco VFrame API SDK, and the other for assessing the *Cisco VFrame Data Center 1.1 Online Programmer's Reference*.

Security Mechanism

Each SOAP session with the VFrame API has its credentials transmitted using HTTP basic authentication. For inbound calls, the username field is formatted similarly to the VFrame Data Center GUI as `<username>@<context>`. These credentials are used to enforce the same role based access

control (RBAC) used by the GUI. HTTPS is supported to provide encrypted communications between the client and server. Certificates on the VFrame Data Center Server are generated automatically, however, certificate-based authentication is not supported through the API.

Session Management

The credentials supplied by the client are authenticated once when the session is first established. Subsequent requests using the same session are not re-authenticated because the VFrame Data Center Server uses cookie based session management. For this reason, the client must be able to support cookies - for security, the session times out after a period of inactivity.

Synchronous and Asynchronous Functionality

When a VFrame API client initiates an operation that can be processed immediately, the client is informed right away whether the operation completed successfully. Some operations may require an extended period of time to complete. When a VFrame API client initiates an operation that the VFrame Data Center Server cannot complete immediately, the VFrame API will return a JobReference value as soon as the job is started. The VFrame API client can check on the status of the operation using the Job Service API. Alternatively, the VFrame API client can wait for a Job Completion Notification.

Error Handling

Error Handling As part of the VFrame Data Center API exception handling, all faults return an error codes and text details to the client. The API shares these error codes through message mapping schemes and infrastructure used by the VFrame Data Center Server.

Web services and service-oriented architecture provide a platform-independent way of leveraging this particular functionality. Exceptions that occur in the web services also are communicated in a platform-independent manner. To accomplish this, exceptions raised by the web services are compliant with the standard SOAP specification.

SOAP Exceptions (also known as SOAP Faults) have the following properties:

- Message - Contents of the exception
- Code - Enum constant that specifies the type of Fault code
- Actor - URL of the Web service method where the exception has occurred
- Detail - Communicates (if used) more information about the exception to the caller



CHAPTER 2

API Functions

This chapter contains information about the functionality provided by the VFrame API. The sections within this chapter cover the following:

- General overview of API capabilities and functions
- API object structure
- A description of each API function - inputs, outputs, and faults.

API Overview

As stated in the Overview, the VFrame API is not intended to provide all of the functionality of the VFrame Data Center GUI. The VFrame API focuses on providing third-party management applications the ability to programmatically manage and monitor services offered by VFrame Data Center. The user is expected to use the VFrame Data Center GUI to preform basic configuration. Then, make use of the VFrame API to perform certain general operations

The functionality exposed by the VFrame API is listed below and organized by functional groups.

**Note**

Additional functionality will be added in future releases.

**Note**

For more information about any of the management topics described below, see the VFrame Data Center online help.

AboutVFrame

VFrame API retrieves information from VFrame Data Center regarding its version:

- Display the version of the VFrame Data Center software currently running

Credentials Management

Device credentials are the usernames, passwords, and SNMP (version 1 or 2) community strings required to log into a device and obtain information that VFrame requires to discover and manage devices. The username and password is the same user account information used to log into the device's CLI to perform system configuration. VFrame API provides the following credential management:

- List Device Credentials
- Add Credentials
- Modify Credentials
- Remove Credentials

Resource Management

Resources (networking devices) are found by running a discovery. Discovered devices appear in VFrame Data Center within the device selector under the Resources tab. The VFrame API allows you to perform the following operation regarding network resources within the data center:

- Query Resources
- Query Resource Pools
- Manage Resources
- Un-manage Resources
- Maintain Resources
- Label Resources

- Update Resource Pools
- Query Physical to Logical Mappings

Service Template Management

Service network templates are made up of logical elements, such as firewall service modules, switches, load balancers, server groups, and so on. Service templates also contain endpoint links. Service network templates act as a network map that includes the elements required for a particular service. Information is provided that defines the template and the elements within. Note that this basic information, along with other information supplied later, is used to create and start a service network.

VFrame API can list defined service templates on the VFrame Data Center Server. In addition, VFrame API can be used to download and upload services template definitions between a local file system and the VFrame Data Center Server:

- List Templates
- Export Logical Template
- Import Logical Template

Service Network Design

In a data center, one might need to create several service networks with only a few differences. However, this task typically requires the creation of each service network, and repeating the same steps and commands for each. VFrame Data Center simplifies this task by allowing you to create a service network based on a published template. Variables are used to fill in the unique service template values required for the devices in each specific service network.

VFrame API can list the current service networks and their elements, which are defined on the VFrame Data Center Server:

- List Service Networks
- List Service Network Elements

Service Operations

VFrame API provides several functions for working with service networks on the VFrame Data Center appliance. These include starting and stopping service networks, adding components to service networks, and querying service networks:

- Start Service Network
- Stop Service Network
- Verify Service Network
- Add Logical Servers
- Image Logical Servers
- Remove Logical Servers
- List Service Element Events
- Execute Service Element Event

- Query Logical to Physical Mappings

Job Management

The VFrame Data Center Jobs API allows third party applications to perform job related operations.

Most actions that are performed using VFrame Data Center are considered jobs, and allow for asynchronous execution. As a result, most of the VFrame API functions (for example, ServiceOperations) return a job reference, which is of type ManagedObjectReference:

- Get Job Runs

Notifications Management

VFrame provides the ability to send asynchronous notifications to external applications, which are considered registered receivers. These notifications are intended to provide status of system events, faults, and so on. To receive these notifications, a receiver must implement a SOAP endpoint using HTTPS transport.

VFrame Notifications are based on a publish/subscribe asynchronous messaging model. This is a “topic” or “named logical channels” based subscription where the “receiver” receives notifications from channels or topics to which they are subscribed.

VFrame Data Center Notifications API provides several functions for managing notifications, including the ability to specify the recipients of e-mail notifications:

- List Notification Topics
- List Notification Receivers
- Register Notifications Receiver
- Deregister Notifications Receiver

Data Objects

The VFrame API provides access to the various objects being managed by the VFrame Data Center. These objects include physical resources, resource pools, service templates, service networks, and so on. DataObjects represent VFrame Data Center business data and they hold their data in properties. A DataObject contains a set of named properties, each of which contains either a simple data-type value or a reference to another DataObject.

Managed Object Reference

Each managed object is uniquely referenced by a ManagedObjectReference or one of its extension subtypes. A ManagedObjectReference contains one optional attribute named objectId of type ManagedObjectID. A ManagedObjectID has two string attributes, “type,” and “value.”

Following, shows the XML schema definition for ManagedObjectID and ManagedObjectReference:

```
<complexType name="ManagedObjectId">
  <sequence>
    <element name="type" minOccurs="1" type="xsd:string" />
    <element name="value" minOccurs="1" type="xsd:string" />
  </sequence>
</complexType>

<complexType name="ManagedObjectReference">
  <sequence>
    <element name="managedObjectId" minOccurs="0"
      type="types:ManagedObjectId" />
  </sequence>
</complexType>
```

The ManagedObjectID “value” attribute is a system generated value that uniquely identifies the managed object.

ManagedObjectIDs and ManagedObjectReferences should be retrieved through query calls by the VFrame API. They may then be passed back to the VFrame API in subsequent calls that require object references. ManagedObjectIDs can be stored on the client for future use, however, it is recommended that the client verify that the managed object still exists before using a stored ManagedObjectID.

For managed object types that are naturally identified by names, the VFrame API uses extensions to ManagedObjectReferences. These extensions define one or more optional attribute that uniquely identifies the managed object. For example, a service template may be identified by a template name. This allows the client to create a ServiceTemplateReference and specify the templateName rather than querying for the ServiceTemplateReference.

Since the objectId attribute inherited from the ManagedObjectReference is optional, extension types of ManagedObjectReference generally contain either the objectId attribute or the extension specific attributes, but not both. ManagedObjectReference instances returned by calls by the VFrame API (even if they are extensions to ManagedObjectReference), always use the objectId attribute and not the extension specific attributes. Extension specific attributes are only used by clients to create object reference instances without specifying the ManagedObjectID value.

For example, when the client queries for all service templates, it receives a collection of ServiceTemplateInfo instances. Each ServiceTemplateInfo instance contains a ServiceTemplateReference that has the ManagedObjectID value. These ServiceTemplateReference instances do not contain the template name. If the client wants to query for all the service networks for

a service template, it passes as an argument a `ServiceTemplateReference` instance that it retrieved earlier. Alternatively, if the client knows the template name, it may directly construct a `ServiceTemplateReference` instance and specify the `templateName` value.

Composite Data Objects

Composite data objects are objects used to access or modify VFrame business data through web services. They are defined using the XML schema. A full listing of all the composite data objects used by the VFrame API is included in the *Cisco VFrame Data Center 1.1 Online Programmer's Reference*.

Faults Objects

In typical Web Services behavior, an exception thrown by the Web Service end point is passed on to the client as a SOAP fault. The fault data objects contain information regarding errors or exceptions encountered during a particular operation. A full listing of all the Fault data objects used by the VFrame API is included in the *Cisco VFrame Data Center 1.1 Online Programmer's Reference*.

VFFault

VFFaults represent custom exceptions or error conditions on the VFrame Data Center Server. They are based on VFrame `FaultType` and has 2 elements.

- `faultKey` - Identifies the fault category
- `faultMessage` - Detailed text regarding the fault

```
<complexType name="VFFaultType">
  <sequence>
    <element name="faultKey" minOccurs="0" type="xsd:string" />
    <element name="faultMessage" minOccurs="0" type="xsd:string" />
  </sequence>
</complexType>
```

VFAuthorizationFault

VFrame Authorization Faults are SOAP faults that indicates an authorization violation during tasks or operations levels. VFrame Data Center uses Role Based Access control to associate tasks to roles (identities) and privileges. The `faultMessage` element provides information on the nature of violations.

VFrame Authorization Fault type are extended from VFrame based `FaultType` data object.

```
<complexType name="VFAuthorizationFaultType">
  <complexContent>
    <extension base="types:VFFaultType">
      <sequence>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

VFInvalidValueFault

This VFrame SOAP fault indicates input data validation exceptions. The `faultMessage` element provides information on the nature of violations. VFrame Invalid Value Fault type is extended from VFrame base FaultType data object.

```
<complexType name="VFInvalidValueFaultType">
  <complexContent>
    <extension base="types:VFFaultType">
      <sequence>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

API Functions

In the last section basic overviews were provided for the services that the VFrame API supports. In this section, each API function is documented to explain their input and output values, as well as possible API fault messages.

Required Inputs

All VFrame API function calls require the following security-based input parameters for authentication:

Parameters

host	Either the IP address or name of the server on which Cisco VFrame Data Center is running.
username	User name of the administrator authorized to use Vframe Data Center.
password	Password for this instance of VFrame Data Center.
context	Context under which this instance of VFrame Data Center is running.

In session-mode, after these parameters are authenticated for the first time, only cookies are used for subsequent authentication and session management. These parameters are contained within the HTTP header.

AboutVframe

VFrame API retrieves information from VFrame Data Center regarding its version:

getVersion

Inputs

None

Outputs

Release Date

Product Name

Product Version

Error Codes

VFFault

VFInvalidValueFault

VFAuthorizationFault

Device Credentials

Device credential functions allow third party applications to retrieve the currently specified device credentials, update or remove existing device credentials, and add new device credentials. These functions allow the administrators to centrally maintain the device credentials with their own management application and programmatically synchronize VFrame Data Center with a central credentials store. This is especially useful for organizations that have periodic password rotation policies. Credentials are stored in the VFrame database and are consulted whenever the VFrame Data Center needs to communicate with a device, including during device discovery and during service deployment.

These functions allow the client to retrieve and modify standard device credentials, as well as default credentials. Standard device credentials are associated with a particular IP address range, while default credentials are not associated with any address range. Default credentials are tried when all standard credentials of matching IP address ranges have failed.

When the VFrame API is used with credentials, the information shown in [Table 2-1](#) is returned by the `getCredential` function and passed in arguments to the `Add` and `Save Device Credential` functions. The `Remove Device Credential` function requires only the `Credential Reference` (`ManagedObjectReference`).

Table 2-1 *Credential Information*

Credential Reference	ManagedObjectReference - a value that identifies the device
Credential Type Code	String identifying the credential type. Current types are: <ul style="list-style-type: none"> • CLI • SNMP v1/v2 • LOMManager • LOM • NFSServer • SANManager
Credential Category Code	The credential category code is another classification of credentials based on device categories. Current values are: <ul style="list-style-type: none"> • Network • NetworkService • Server • Storage <p>NOTE: there is a dependency between Credential category code and Credential type code.</p>
IP Address Range	A list of one or more IP addresses – A null or empty range is possible for only default credentials
Username	The credential username – This field is ignored for SNMP v1/v2 credentials
Password	The credential password – This field is ignored for SNMP v1/v2 credentials
Enable Password	The credential password with authorization to make configuration changes – This field is ignored for SNMP v1/v2 credentials

Table 2-1 *Credential Information (continued)*

Read Community	The SNMP v1/v2 read community string – Used only for SNMP v1/v2 credentials and is ignored for all other credential types
Write Community	The SNMP v1/v2 write community string – Used only for SNMP v1/v2 credentials and is ignored for all other credential types

getCredentials

This API function allows the client to retrieve credentials configured on VFrame Data Center. The client can retrieve one or more credentials by specifying credential type, by credential reference, or both.

Inputs

ManagedObjectReference identifying the credential

CredentialTypeCode

CredentialCategoryCode

Outputs

List of credential instances

Error Codes

VFFault

VFInvalidValueFault

VFAuthorizationFault

addCredentials

This API function allows the client to add new credentials to the VFrame Data Center's database. The provided credential reference is ignored. The newly created credentials will be returned with the actual ManagedObjectReference for the new credentials.

Inputs

List of credential instances

Outputs

List of credential instances with valid ManagedObjectReference

Error Codes

VFFault

VFInvalidValueFault

VFAuthorizationFault

saveCredentials

This API function allows the client to save changes to existing credentials - the updated credentials are returned. The provided credential instance's ManagedObjectReference must match that of an existing credential, otherwise the credential is ignored.

Inputs

List of credential instances with changes

Outputs

List of credential instances

Error Codes

VFFault
VFInvalidValueFault
VFAuthorizationFault

removeCredentials

This API function allows the client to remove existing credentials from VFrame Data Center's database.

Inputs

List of ManagedObjectReference instances identifying the credential to be removed

Outputs

None

Error Codes

VFFault
VFInvalidValueFault
VFAuthorizationFault

Resources

Resources include physical resources such as servers and network devices as well as logical resources such as IP addresses and VLANs. VFrame resource APIs provide a way to list the resources known to VFrame Data Center and to retrieve the resource physical to logical mappings. Also, resources can be placed in a Managed or Unmanaged states through the resource APIs.

Table 2-2 lists the properties of a ResourceInfo instance:

Table 2-2 ResourceInfo Properties

Resource Reference	ManagedObjectReference identifying the resource
Name	A string identifier (e.g., IP Address), which can be used to represent the resource in a UI
IsNameSetByUser	Boolean flag indicating whether the name is a default system generated name – If true, the name is a user given name.
Resource Type Code	String code identifying the resource type – This value maps to the Resource Type Code from the Resource Type instances returned in the GetAllResourceTypes operation
State	The current state of the resource – Currently, supported states are: New, Managed, Unmanaged, Failed, Verifying, and Unsupported
Managed Sub State	The sub state of managed resources – Currently supported sub states are: Not in Use, Config Failed, Available In Use, Saturated, and Maintenance
Attributes	<p>Attributes specific to the resource type. These attributes can be used to correlate the resource to its identity as known by a third party application. The list of specific attributes available for each type is specified in the resource type structure. The attributes are returned as sets of attribute name/value pairs.</p> <p>A resource pool has the following attributes:</p> <ul style="list-style-type: none"> • ResourcePoolReference, which identifies the particular resource pool • Name - A user configured name • Description - a user configured description • ResourceTypeCode - the type of resource contained in the pool – A pool contains resources only of the same type • System Generated Flag – A flag indicating whether this pool was system generated

getResourcePools

This API function allows the client to query the available resource pools. This can be used for reporting purposes. For example, the administrator may be interested in tracking utilization of a particular pool. After retrieving the available pools, the client can retrieve the resources (by resource pool) to generate reports. The VFrame API client can retrieve the pools all at once, by resource type, or by ResourcePoolReference.

Inputs

Resource Type

ResourcePoolReference

Outputs

List of resource pool instances

Error Codes

VFFault

VFInvalidValueFault

VFAuthorizationFault

getResources

This API function allows the client to retrieve the actual resource instances known to the VFrame Data Center. The client can retrieve all the resources at once, by reference, type, pool, state, or by managed sub state.

Inputs

Resource Type Code

ResourcePoolReference

Resource State

Resource Managed Sub State

ManagedObjectReference of the resource

Outputs

List of resource instances

Error Codes

VFFault

VFInvalidValueFault

VFAuthorizationFault

labelResources

Every resource has a system defined default name. Use this API function to modify the resource name and description. Whenever a resource name is set by the user, the resource attribute `IsNameSetByUser` is set to true.

Table 2-3 shows the attributes of a `ResourceLabelInfo` instances:

Table 2-3 *ResourceLabelInfo Instance*

ManagedObjectReference	ResourcePoolReference identifying the resource pool
Name	A user configured name
Description	A user configured description (optional)

Inputs

List of `ResourceLabelInfo` instances

Outputs

List of updated `ResourceLabelInfo` instances

Error Codes

VFFault

VFInvalidValueFault

VFAuthorizationFault

maintainResources

This VFrame API function is used to place a device into resource maintenance mode.

If the input is `MoveToMaintenance`, the device is moved to maintenance mode, which allows any type of maintenance task to be performed on the device, such as physical device repair, device hardware replacement, or software upgrade. This changes the device state to *Resource Under Maintenance*. If the `MoveToMaintenance` input is entered while the device is being used, the device is tagged as *Resource Marked for Maintenance*. When the device is no longer in use, the device state changes to *Resource Under Maintenance*.

Once maintenance is complete, use the `MaintenanceCompleted` input. This changes the device state to *Managed* and is once again available for use.

`CancelMaintenance` input cancels maintenance for a device that is in the *Marked for Maintenance* state.

Inputs

A list of ManagedObjectReference instances identifying the resources to be maintained and the maintenance operation code. The operation code should be one of the following:

- MoveToMaintenance
 - MaintenanceCompleted
 - CancelMaintenance
-

Outputs

None

Error Codes

VFFault
VFInvalidValueFault
VFAuthorizationFault

manageResources

This API function places a set of resources in the managed state. VFrame Data Center can connect to the specified resources to verify they are ready to be managed. This function also can collect additional inventory information and make configuration changes. This function is performed asynchronously in a job, which implies that the return response may not occur immediately.

Inputs

A list of ManagedObjectReference instances that identify the resources to be managed

Outputs

JobReference that identifies the job

Error Codes

VFFault
VFInvalidValueFault
VFAuthorizationFault

unManageResources

This API function is the opposite of the manageResources function. It places a set of resources into the unmanaged state. VFrame Data Center may connect to the specified resource to query additional information or make configuration changes. This function is performed asynchronously in a job, which implies that the return response may not occur immediately.

Inputs

A list of ManagedObjectReference instances that identify the resources to be unmanaged

Outputs

JobReference that identifies the job

Error Codes

VFFault
 VFInvalidValueFault
 VFAuthorizationFault

getResourcePoolDetails

This API function is similar to the getResourcePools operation, except this function returns the resource pool identification information, as well as the resources in the pool. The returned resource pool details can then be modified, including the pool membership. Once this is done, updated resource pool information can be sent to the VFrame Data Center Server with the saveResourcePools operation.

A Resource Pool Detail instance contains all the fields from the Resource Pool instance plus a list of resource instances for the resources in the pool.

Inputs

Resource Type
 ResourcePoolReference

Outputs

A list of Resource Pool Detail Instances

Error Codes

VFFault
 VFInvalidValueFault
 VFAuthorizationFault

saveResourcePools

This API function saves modifications to resource pools to the VFrame Data Center's database. This function can change the pool name, description, and pool membership.

Inputs

A List of modified Resource Pool Detail instances

Outputs

A list of the Updated Resource Pool Details

Error Codes

VFFault
VFInvalidValueFault
VFAuthorizationFault

getLogicalElementsForResource

This API function returns the logical resources that are assigned to or required by a physical resource.

Inputs

A ManagedObjectReference that identifies the resource

Outputs

A list of Service Element instances to which the physical resource is assigned to or has been acquired by

Error Codes

VFFault
VFInvalidValueFault
VFAuthorizationFault

Service Templates

The Service Template APIs allow the client to view, export, and import service templates. DataObject ServiceTemplateInfo is used to represent an instance of a ServiceTemplate.

getServiceTemplates

This API function lists all of the service templates in the VFrame Data Center's database.

DataObject ServiceTemplateInfo is used to represent a ServiceTemplate. A ServiceTemplateInfo instance has the following attributes:

Table 2-4 *ServiceTemplateInfo Instance*

ref	ReferenceObject identifying the template
templateName	Name of template to which service belongs
templateState	State of the template
allowMacros	Boolean indicating if Macros are allowed for this Template

Inputs

 None

Outputs

 A list of Service Template instances

Error Codes

 VFFault
 VFInvalidValueFault
 VFAuthorizationFault

exportServiceTemplate

This API function allows a specified service template to be exported and displayed or saved to a file system.

Inputs

 Reference to Service Template

Outputs

 A byte array of the exported content

Error Codes

 VFFault
 VFInvalidValueFault
 VFAuthorizationFault

importServiceTemplate

This API function allows previously exported service templates to be imported into the VFrame Data Center's database. If a name collision occurs, then such names are renamed and a list of renamed elements is returned.

Inputs

 Name of the XML file to be imported

Outputs

 Renamed element list

Error Codes

VFFault
 VFInvalidValueFault
 VFAuthorizationFault

Service Design

The Service Design APIs allow the client to define new service networks. These APIs are used for reporting purposes, as well as to retrieve the identifiers to use in the Service Operations API functions. Associated with a Service Network are one or more Service Elements. The Service Element represents a logical element in the Service Network. DataObject ServiceElementInfo is used to represent an instance of a ServiceElement. DataObject ServiceNetworkInfo is used to represent an instance of a ServiceNetwork.

getServiceNetworkList

This API function is used by the client to query the service networks currently defined. The client may retrieve the service networks by template reference or service network reference.

DataObject ServiceNetworkInfo is used to represent a Service Network. A ServiceNetworkInfo instance has the following attributes:

Table 2-5 *ServiceNetworkInfo Instance*

ref	ManagedObjectReference of service network
templateName	Name of template to which service belongs
serviceName	Name of the service network
designState	Design state of the service network
operationState	Operational state of the service network

Inputs

Reference of ServiceTemplate for which all of the Service Networks are to be listed

Outputs

A list of Service Networks

Error Codes

VFFault
 VFInvalidValueFault
 VFAuthorizationFault

getServiceElements

This API function queries the available Service Element instances. The Service Elements can be queried by service network or by element reference. Since Service Element instances also contain their sub elements, this function returns the entire element tree.

DataObject ServiceElementInfo is used to represent a Service Network. Note that a service element can contain other child service elements. For example, a ServerGroup service element contains the *servers* as the child service elements.

A ServiceElementInfo instance has the following attributes:

Table 2-6 *ServiceElementInfo Instance*

ref	ManagedObjectReference of service element
templateName	Name of template to which service belongs
serviceName	Name of the service network
elementName	Name of the service element
operationState	Operational state of the service element
faultState	Fault state of the service element
subElements	Array of child service elements

Inputs

Reference of Service Network for which all Service Elements are to be listed

Outputs

A list of Service Elements

Error Codes

VFFault
VFInvalidValueFault
VFAuthorizationFault

Service Operations

The Service Operations API functions allow third party applications to dynamically start, stop, maintain, and resume previously defined services.

startServiceNetwork

Use this API function to start a service.

Inputs

Service Network Reference

Outputs

JobReference (ManagedObjectReference)

Error Codes

VFFault

VFInvalidValueFault

VFAuthorizationFault

stopServiceNetwork

Use this API function to deactivate a service.

Inputs

Service Network Reference

Outputs

JobReference (ManagedObjectReference)

Error Codes

VFFault

VFInvalidValueFault

VFAuthorizationFault

verifyServiceNetwork

Use this API function to verify a service.

Inputs

Service Network Reference

Outputs

 JobReference (ManagedObjectReference)

Error Codes

 VFFault

VFInvalidValueFault

 VFAuthorizationFault

createLogicalServer

This API function adds n servers to the ServerGroup service element whose reference is provided as input.

Inputs

 ManagedObjectReference (of the server group to which the server is added)

 Number of servers to be created

Outputs

 JobReference

Error Codes

 VFFault

VFInvalidValueFault

 VFAuthorizationFault

imageServers

This API function accepts an array of logical servers and replicates the golden image on each of them.

Inputs

 ArrayOfManagedObjectReference

Outputs

 JobReference

Error Codes

 VFFault

VFInvalidValueFault

 VFAuthorizationFault

deleteLogicalServers

Use this API function to delete the servers from a server group.

Inputs

ArrayOfManagedObjectReference

Outputs

JobReference

Error Codes

VFFault
VFInvalidValueFault
VFAuthorizationFault

getResourceMappingForServiceElement

This API function returns the physical resources assigned or acquired by a given logical service element.

Inputs

ServiceNetworkReference or
ManagedObjectReference

Outputs

A list of physical resource instances acquired by the
given service network or service element

Error Codes

VFFault
VFInvalidValueFault
VFAuthorizationFault

getServiceEvents

This API function lists all the events defined for the specified service element.

DataObject ServiceElementEventInfo is used to represent a Service Element Event. A ServiceElementEventInfo instance has the following attributes:

Table 2-7 *ServiceElementEventInfo Instance*

eventId	ManagedObjectReference for the event
eventName	Name of the event
eventType	Type of event

Table 2-7 *ServiceElementEventInfo Instance (continued)*

selectionState	Lifecycle state to which the event is applicable
isUserDefined	Boolean indicating if the event is user defined
isVisibleOperation	Boolean indicating if the event is visible to users
Description	Description of the event

Inputs

ManagedObjectReference

Outputs

ArrayOfServiceElementEventInfo

Error Codes

VFFault
VFInvalidValueFault
VFAuthorizationFault

executeServiceElementOperation

Use the API function to execute an event.

Inputs

ManagedObjectReference

Outputs

JobReference

Error Codes

VFFault
VFInvalidValueFault
VFAuthorizationFault

Jobs

The VFrame Data Center Jobs API allows third party applications to perform job related operations. Most actions that are performed using VFrame Data Center are considered jobs, and allow for asynchronous execution. As a result, most of the VFrame API functions (for example, ServiceOperations) return a job reference, which is of type ManagedObjectReference.

**Note**

From the VFrame Data Center GUI, while in “Admin” context, you can see a consolidated view of jobs by selecting View > Jobs.

**Note**

The current release of the VFrame API provide operations only to query job information.

A JobInfo DataObject represents the basic definition of the job, itself. The JobInfo instance has the following fields:

Table 2-8 JobInfo Instance

ref	ManagedObjectReference of the JobInfo
type	JobType, e.g., discovery, imaging, etc.
Name	Name of the Job
params	Parameters used in the Job
recurring	Boolean indicating a recurring Job
nextRun	Timestamp of next job run instance
lastUserModifiedTime	Job last updated time stamp
lastModifiedBy	Job last modified by

When a job is executed, the details are provided as a JobRunInfo data object. A Job (i.e., JobInfo) has at least one JobRunInfo instance. Only recurring jobs have multiple JobRunInfo instances.

The JobRunInfo instance has the following fields:

Table 2-9 JobRunInfo Instance

runref	ManagedObjectReference of the JobRunInfo
jobRef	ManagedObjectReference of the parent JobInfo
status	Status of the job executed
extendedStatus	Extended Status (if any) of the job executed
startTime	The job run start time
nextRun	Timestamp of next job run instance
endTime	The job run end time
percentComplete	How much of the job has finished
log	Array of JobLog data object

JobLogEntry provides more details regarding the Job that was executed. Errors and warnings from the execution are contained in the job log. A JobRunInfo instance can have one or more JobLogEntry instances.

The JobLogEntry instance has the following fields:

Table 2-10 JobLogEntry Instance

timestamp	Date and time that the message was logged
severity	Severity level of the message
status	Status of the job executed
message	The job log message

The Job query criteria is specified using the JobRunQuery DataObject.

A JobRunQuery instance has the following fields:

Table 2-11 JobRunQuery Instance

jodRef	ManagedObjectReference of the JobRunInfo
runRef	ManagedObjectReference of the parent JobInfo
typeCode	Type of the job
runStatusCode	Status of the job
runStartTimeFrom	The job run start time begin range
runStartTimeTo	The job run start time end range
runEndTimeFrom	The job run end time begin range
runEndTimeTo	The job run end time end range
maxResults	Indicates maximum number of Job to be returned - range is 1 to 50

getJobRuns

This API function retrieves the details for a particular job.

VFrame API clients can use this operation to poll for job status. The query criteria is specified using the JobRunQuery DataObject. Leave out any parameters that should be ignored. For example, do not include the “jobRef” element if not querying for a specific job. Conflicting parameters may result in zero jobs being returned. If a JobRef is provided, but the job has not been run, no JobRunInfo elements are returned, but the JobInfo for the Job is returned. A JobRunsReturn DataObject contains the output of the operation.



Note

JobRunsReturn is made up of JobInfo and JobRunInfo.

Inputs

JobRunQuery instance

Outputs

 JobRunsReturn instance

Error Codes

 VFFault

VFInvalidValueFault

 VFAuthorizationFault

Notifications

VFrame Data Center provides the ability to send asynchronous notifications to registered receivers, which are external applications. These notifications are intended to provide status of system events, faults, and so on. To receive these notifications, a receiver must implement a SOAP endpoint using HTTPS transport.

VFrame Notifications are based on a publish/subscribe asynchronous messaging model. This is a “topic” or “named logical channels” based subscription, where the “receiver” receives notifications from channels or topics to which they are subscribed.

Notification receiver instances are returned from list operations, and passed in when registering for notifications. [Table 2-12](#) shows the information contained in a notification receiver instance:

Table 2-12 Notification Receiver Instance

Notification Receiver Reference	NotificationReceiverReference identifying the notification receiver.
URI	The address of the SOAP endpoint – Only HTTP or HTTPS URLs are supported
Username	Username for basic authentication
Password	Password for basic authentication
Version	Version of the VFrame API implemented by the receiver
Topic	List of one or more TopicInfo

A TopicInfo DataObject contains information about the topic to which one is subscribed or unsubscribed. [Table 2-13](#) shows the information contained within TopicInfo instance.

Table 2-13 TopicInfo Instance

TopicName	Name of the topic to subscribe to or unsubscribe from.
ManagedObjectReference	Reference of the Managed Object for which the given topic applies – this is required by some of the topics

getTopics

This API function returns a list of all the Topics to which a receiver can subscribe.

Inputs

None

Outputs

List of Topics and their descriptions

Error Codes

VFFault

VFInvalidValueFault

VFAuthorizationFault

Available Topics

The following is a list of currently supported topics:

SERVICE_TEMPLATE_STATE_CHANGE – Topic for Logical Template Change Notification.

SERVICE_NETWORK_STATE_CHANGE – Topic for Logical Network Change Notification.

SERVICE_ELEMENT_STATE_CHANGE – Topic for Logical Element Change Notification. Requires ManagedObjectReference of the Service Network to which the Service Element belongs, as an additional parameter.

JOB_COMPLETED – Topic for Job Completed Notification. Requires the scheduled Job's ManagedObjectReference as an additional parameter.

SERVER_INVENTORY_FINISHED – Topic for “Server Inventory Finished” Notification.

LOGICAL_SERVER_CREATED – Topic for Logical Server Add Notification. Requires ManagedObjectReference of the ServerGroup for which the server is created as an additional parameter.

LOGICAL_SERVER_DELETED – Topic for Logical Server Delete Notification. Requires ManagedObjectReference of the ServerGroup for which the server is deleted as an additional parameter.

FAULTS_CHANGE – Topic for Fault Change Notification.

RESOURCE_LIFECYCLE_STATE_CHANGE – Topic for Resource lifecycle state changed Notification.

RESOURCE_MANAGED_STATE_CHANGE – Topic for Resource Managed State Notification.

RESOURCE_FAULT_STATE_CHANGE – Topic for Resource Fault State Notification.

registerReceiver

This API function registers an endpoint to receive notifications.

Inputs

Notification Receiver instance

Outputs

None

Error Codes

VFFault
VFInvalidValueFault
VFAuthorizationFault

unRegisterReceiver

This API function has two modes:

- It can be used to completely remove (un-register) a notification receiver endpoint from the list of notification receivers or just unsubscribe a particular topic.
- If only a Notification Receiver ManagedObjectReference is provided, the notification receiver is deleted along with all the topics to which the receiver is subscribed. However, if topic information is provided along with the notification receiver reference, then only that topic is unsubscribed for that notification receiver.

Inputs

ManagedObjectReference
TopicInfo (optional)

Outputs

None

Error Codes

VFFault
VFInvalidValueFault
VFAuthorizationFault

listReceivers

This operation lists the registered notification receivers for the context of the calling user, or from all contexts if the user is from the “Admin” context.

Inputs

None

Outputs

List of NotificationReceiver instances

Error Codes

VFFault

VFInvalidValueFault

VFAuthorizationFault

Notification Reply Information

Depending on the receiver types and topics, there can be a number of different outcomes to a notification configuration. This section describes the types of notification and the information contained within their replies. Keep in mind that these notifications can be synchronous (giving an immediate reply) or asynchronous (sent only when a particular event has occurred).

Notification Details

When an asynchronous operation (event) is completed, VFrame Data Center sends an instance of NotificationInfo to all the receivers who have subscribed to that topic.

The NotificationInfo instance contains the following details:

Notification Type

Notification Type is the name of the topic related to the event

Date

Date and Time when the notification was sent by VFrame Data Center.

Data

This represents the actual notification contents and is in the form of a list of properties – Each property is a name/value pair, the list of properties in a NotificationInfo depends on the topic

Service Template State Change Notification

A Service Template State Change event is triggered whenever there is a change to the State of any Service Templates defined within VFrame Data Center. Receivers subscribed to this topic, receive the following NotificationInfo details.

Notification Type

SERVICE_TEMPLATE_STATE_CHANGE

Date

Timestamp when the notification was generated by VFrame Data Center

Data

Property Name	Property Value
ManagedObjectId	ManagedObjectId value of the affected Service Template
TargetType	ServiceTemplate
Action	StateChange
OldState	Old state information
NewState	New state information
Message	A brief description of the event

Service Network State Change Notification

Service Network State Change events are triggered whenever there is a change in the State of any Service Networks defined in the VFrame Data Center's database. Receivers subscribed to a Service Network topic that changes state, receive the following NotificationInfo details.

Notification Type

SERVICE_NETWORK_STATE_CHANGE

Date

Timestamp when the notification was generated by VFrame Data Center

Data

Property Name	Property Value
ManagedObjectId	ManagedObjectId value of the affected Service Network
TargetType	ServiceNetwork
Action	StateChange
OldState	Old state of the service network
NewState	New state of the service network
OldOperationState	Old operation state of the service network

Property Name	Property Value
NewOperationState	New operation state of the service network
Message	A brief description of the event

Service Element State Change Notification

Service Element State Change events are triggered whenever there is a change to the State of any service element.

Note that registering for this topic requires the ManagedObjectReference of the service network to which the service elements belong. Receivers subscribed to this topic receive the following NotificationInfo details.

Notification Type

SERVICE_ELEMENT_STATE_CHANGE

Date

Timestamp when the notification was generated by VFrame Data Center

Data

Property Name	Property Value
ManagedObjectId	ManagedObjectId value of the affected Service Element
TargetType	Service Element
Action	StateChange
OldState	Old state of the service element
NewState	New state of the service element
Message	A brief description of the event

Fault Change Notification

VFrame Data Center issues fault alarms if a problem occurs with a managed physical resource or an operational virtual resource (such as a device or service network).

Receivers subscribed to this topic receive the following NotificationInfo details.

Notification Type

FAULT_CHANGE

Date

Timestamp when the notification was generated by VFrame Data Center

Data

Property Name	Property Value
ManagedObjectId	ManagedObjectId value of the FaultAlarm
TargetType	ManagedObject
Action	Fault
State	Fault state
Severity	Severity of the fault
Message	A brief description of the event

Job Completed Notification

Note that registering for this topic requires the ManagedObjectReference of the JobRun, which is included in the JobRunsReturn DataObject returned by all operations executing in Asynchronous mode. For example, Create, Image, and Delete server operations return JobRunsReturn data objects as their response.

Receivers subscribed to this topic will receive the following NotificationInfo details.

Notification Type

JOB_COMPLETED

Date

Timestamp when the notification was generated by VFrame Data Center

Data

Property Name	Property Value
ManagedObjectId	ManagedObjectId value of the Job
TargetType	Job
Action	JobComplete
Status	Status of the Job
Message	A brief description of the event

Server Inventory Finished Notification

A Server Inventory Finished event is triggered whenever a physical server inventory or re-inventory is completed. Receivers subscribed to this topic receive the following NotificationInfo details.

Notification Type

SERVER_INVENTORY_FINISHED

Date

Timestamp when the notification was generated by VFrame Data Center

Data

Property Name	Property Value
ManagedObjectId	ManagedObjectId value of the affected Physical Server
TargetType	PhysicalServer
Action	Inventory
State	Fault state
Message	A brief description of the event

Logical Server Created Notification

Note that registering for this topic requires the ManagedObjectReference of the server group within which the logical server is created.

A Logical Server Created Notification is sent whenever a logical server is created. One notification is sent for each server that is created.

Notification Type

LOGICAL_SERVER_CREATED

Date

Timestamp when the notification was generated by VFrame Data Center

Data

Property Name	Property Value
ManagedObjectId	ManagedObjectId value of the newly created logical server
TargetType	LogicalServer
Action	Added
Message	A brief description of the event

Logical Server Deleted Notification

A Logical Server Deleted Notification is sent whenever a logical server is deleted. One notification is sent for each server that is deleted.

Note that registering for this topic requires the ManagedObjectReference of the server group from which the logical server is deleted.

Notification Type

LOGICAL_SERVER_DELETED

Date

Timestamp when the notification was generated by VFrame Data Center

Data

Property Name	Property Value
ManagedObjectId	ManagedObjectId value of the deleted logical server
TargetType	LogicalServer
Action	Deleted
Message	A brief description of the event

Resource LifeCycle State Change Notification

Resource LifeCycle State Change events are triggered whenever there is a change to the LifeCycle state of a any resource. For example, whenever a resource is managed or unmanaged, a LifeCycle State Change events is triggered. Receivers subscribed to this topic receive the following NotificationInfo details.

Notification Type

SERVICE_TEMPLATE_STATE_CHANGE

Date

Timestamp when the notification was generated by VFrame Data Center

Data

Property Name	Property Value
ManagedObjectId	ManagedObjectId value of the affected Resource
TargetType	Resource
Action	StateChange
OldState	Old state information
NewState	New state information
Message	A brief description of the event

Resource Managed State Change Notification

A Resource Managed State Change event is triggered whenever there is a change to the managed substate of a any resource. For example, whenever a resource is acquired or released by a service element. Receivers subscribed to this topic receive the following NotificationInfo details.

Notification Type

SERVICE_MANAGED_STATE_CHANGE

Date

Timestamp when the notification was generated by VFrame Data Center

Data

Property Name	Property Value
ManagedObjectId	ManagedObjectId value of the affected Resource
TargetType	Resource
Action	StateChange
OldState	Old state information
NewState	New state information
Message	A brief description of the event

Resource Fault State Change Notification

Resource Fault State Change events are triggered whenever there is a change to the fault state of a any resource. Receivers subscribed to this topic receive the following NotificationInfo details.

Notification Type

SERVICE_FAULT_STATE_CHANGE

Date

Timestamp when the notification was generated by VFrame Data Center

Data

Property Name	Property Value
ManagedObjectId	ManagedObjectId value of the affected Resource
TargetType	Resource
Action	StateChange
OldState	Old state information
NewState	New state information
Message	A brief description of the event



CHAPTER 3

VFrame Data Center API SDK

The VFrame API SDK provides the VFrame Data Center with the following development components:

- WSDL files containing the VFrame API service definitions. These files are used to generate client code for communicating with the VFrame Data Center through the VFrame APIs. The client can use any web services toolkit on any platform.
- Sample VFrame API client code, using Apache Axis 1.3. The sample code can be used as reference or copied and used directly in other clients. The sample client also contains a utility to capture the messages sent and received by the client.
- Apache ANT build script for generating an Apache Axis 1.3 client source. These ANT build scripts can be used to build the sample VFrame API client. The build scripts can be used as a starting point for creating new VFrame API clients or as a reference for building clients.
- Documentation, including this guide, as well as an online HTML reference guide.

Unpacking the API/SDK Files

From your VFrame client computer's browser, enter the URL `http://<vframe-host>/sdk`, where `<vframe-host>` is the IP address (or name if your network supports DNS) of the VFrame Data Center Server. Here, there are two active links: one for the Cisco VFrame API SDK (where the WSDL files are located), and one for assessing the *Cisco VFrame Data Center 1.1 Online Programmer's Reference*.

After going to the URL listed above (URL `http://<vframe-host>/sdk`), select the link, [Please click here to download Cisco VFrame Data Center SDK](#). Selecting this link allows you to save the SDK .zip file to your VFrame client.

Next, Unzip the files (to some location in your directory structure) using the application applicable for your VFrame client computer:

- On Windows, use WinZip
- On UNIX, use the unzip utility

The VFrame folder is now unpacked and resides on your client's file system.



Note

To view the *Cisco VFrame Data Center 1.1 Online Programmer's Reference* immediately, select the link, [WSDL Documentation](#). A copy of this reference also is contained within the VFrame directory on your VFrame client.

VFrame Directory Contents

The VFrame directory contains three major subdirectories:

- **docs** – Contains a copy of this manual in PDF-format, and the *VFrame Data Center 1.1 Online Programmer’s Reference* – activated by selecting the “*index.html*” file.
- **wsdl** – Contains all of the WSDL service files used by the API. The wsdl directory contains many WSDL and related XSD files. However, the top level WSDL files are listed below. Other WSDL files are referenced (imported) from these main files.
 - credentials-service.wsdl
 - resource-service.wsdl
 - jobs-service.wsdl
 - servicetemplates-service.wsdl
 - servicedesign-service.wsdl
 - serviceoperations-service.wsdl
 - notifications-service.wsdl
 - notificationsReceiver-service.wsdl
 - aboutvframe-service.wsdl
- **sample** – Contains the “java/bin” directory, which contains simple, sample VFrame client utilities. These allow API functions to be run from within a wrapper script or command.

Sample Client Utilities

There are two client utilities (one for Windows, the other for UNIX/LINUX), which allow you to send WSDL commands to the VFrame Data CenterServer. These commands are in the Vframe\sample\java\bin directory and are named **run-vframe-api.bat**, for Windows; and **run-vframe-api.sh**, for UNIX/LINUX. Both of these utilities run from within a command prompt window.

Following, is an example of using **run-vframe-api.bat** (Windows) to perform a “getCredentials” operation on a VFrame Data Center Server:

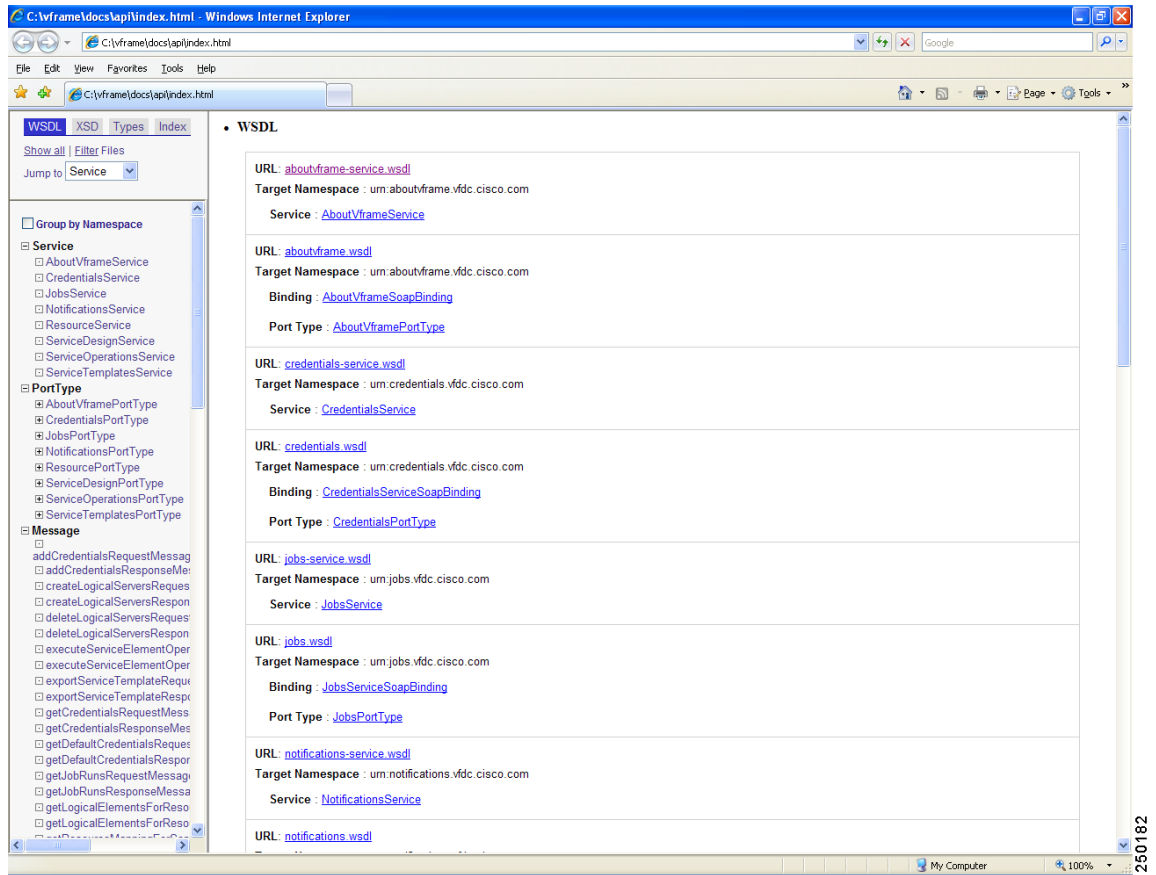
```
C:\>vframe\sample\java\bin>run-vframe-api.bat Credentials -host 10.20.252.10 -user admin
-context AdminContext -password admin -operation List
1441792 | Default | CLI | Network | admin | admin123 | *.*.*.* | null | null
1441794 | Default | CLI | NetworkService | admin | admin123 | *.*.*.* | null | null
1441793 | Default | SNMPv1n2 | Network | null | null | *.*.*.* | public | null
1441797 | Default | NFSServer | Storage | root | admin123 | 20.5.17.3 | null | null
```

Following, is an example of using **run-vframe-api.sh** (UNIX\LINUX) to perform a “getCredentials” operation on a VFrame Data Center Server:

```
>./run-vframe-api.sh Credentials -host 10.20.252.10 -user admin -password admin123
-context AdminContext -operation List -type CLI
1376256| Network | CLI | Network | root | admin123 | *.*.*.* | null | null |
1376263 | arijit | CLI | Network | ari | ari123 | 10.1.1.1 | null | null |
1376257 | network | CLI | NetworkService | root | admin123 | *.*.*.* | null | null |
```

Online Programmer's Reference

The figure below is an example of the opening page of the interactive *Cisco VFrame Data Center 1.1 Online Programmer's Reference*. Contained within the Programmer's Reference are detailed descriptions of all of the XML, XSD, and WSDL code. Here, you can find definitions and substructures for all object contained within the VFrame Data Center API.





INDEX

A

AboutVFrame [2-2](#)
AboutVframe [2-8](#)
Action [2-34, 2-35, 2-36](#)
Actor [1-6](#)
addCredentials [2-10](#)
allowMacros [2-17](#)
Apache ANT build script [3-1](#)
Apache Axis [1-3](#)
Apache Axis 1.x client [3-1](#)
Apache Software Foundation [1-3](#)
API Functions [2-1, 2-8](#)
API Overview [2-2](#)
API Structural Diagram [1-3](#)
Asynchronous Functionality [1-6](#)
Attributes [2-12](#)
Audience [i](#)
authenticated [1-6](#)
Axis Web services framework [1-3](#)

B

Binding Style [1-2](#)
build scripts [3-1](#)

C

Caution [ii](#)
central credentials store [2-9](#)
certificate-based authentication [1-6](#)
Certificates [1-6](#)
Character Encoding [1-2](#)

Cisco VFrame Data Center 1.1 Online Programmer's Reference [iii, 3-4](#)

CLI [2-9](#)
Code [1-6](#)
Commands and keywords [ii](#)
communicating with the VFrame Data Center [3-1](#)
complexType [2-5](#)
Composite Data Objects [2-6](#)
context [2-8](#)
Conventions [ii](#)
cookies [1-6](#)
createLogicalServer [2-22](#)
Credential Category Code [2-9](#)
Credential Reference [2-9](#)
Credentials Management [2-2](#)
Credential Type Code [2-9](#)

D

DataObject [2-5](#)
Data Objects [2-5](#)
data-type value [2-5](#)
definition [2-5](#)
deleteLogicalServers [2-23](#)
Description [2-12, 2-14, 2-24](#)
designState [2-19](#)
Detail [1-6](#)
Device Credentials [2-9, 2-10, 2-11](#)
Documentation [3-1](#)

E

elementName [2-20](#)

empty string [1-2](#)
 Enable Password [2-9](#)
 Encoding Rule [1-2](#)
 endTime [2-25](#)
 Error Handling [1-6](#)
 Error Handling, Actor [1-6](#)
 eventId [2-23](#)
 eventName [2-23](#)
 eventType [2-23](#)
 Exceptions [1-6](#)
 executeServiceElementOperation [2-24](#)
 exportServiceTemplate [2-18](#)
 extendedStatus [2-25](#)
 extensions [2-5](#)

F

faultKey [2-6](#)
 faultMessage [2-6](#)
 Faults Objects [2-6](#)
 faultState [2-20](#)
 Fault State Change [2-36](#)
 FaultType [2-6](#)
 format or specificity [1-2](#)

G

getCredentials [2-10](#)
 getJobRuns [2-26](#)
 getLogicalElementsForResource [2-17](#)
 getResourceMappingForServiceElement [2-23](#)
 getResourcePoolDetails [2-16](#)
 getResourcePools [2-13](#)
 getResources [2-13](#)
 getServiceElements [2-20](#)
 getServiceEvents [2-23](#)
 getServiceNetworkList [2-19](#)
 getServiceTemplates [2-17](#)

getVersion [1-4, 2-8](#)

H

Header element [1-2](#)
 host [2-8](#)
 HTTP [1-2](#)
 http
 //sdk [3-1](#)
 HTTP 500 errors [1-4](#)
 HTTPS [1-6](#)
 HTTPS transport [2-4](#)

I

imageServers [2-22](#)
 importServiceTemplate [2-18](#)
 Interoperability Considerations [1-5](#)
 interoperability requirements [1-5](#)
 Introduction [1-1](#)
 Invalid Value Fault [2-7](#)
 IP Address Range [2-9](#)
 IsNameSetByUser [2-12](#)
 isUserDefined [2-24](#)
 isVisibleOperation [2-24](#)

J

Job Completion Notification [1-6](#)
 JobInfo data object [2-25](#)
 JobInfo Instance [2-25](#)
 JobLogEntry [2-26](#)
 Job Management [2-4](#)
 jobRef [2-25](#)
 JobReference [1-6](#)
 job reference [2-4](#)
 JobRunInfo [2-25](#)
 JobRunQuery [2-26](#)

JobRunQuery data object [2-26](#)
 JobRunQuery Instance [2-26](#)
 Job Runs [2-4](#)
 Jobs [2-24, 2-26](#)
 jodRef [2-26](#)

L

labelResources [2-14](#)
 lastModifiedBy [2-25](#)
 lastUserModifiedTime [2-25](#)
 Log [2-25](#)
 logical elements [2-3](#)
 Logical Server Created [2-34](#)
 Logical Server Created Notification [2-34](#)
 Logical Server Deleted [2-34, 2-35](#)
 Logical Server Deleted Notification [2-34](#)
 LOM [2-9](#)
 LOMManager [2-9](#)

M

maintainResources [2-14](#)
 ManagedObjectID [2-5](#)
 ManagedObjectId [2-34, 2-35, 2-36](#)
 ManagedObjectIDs [2-5](#)
 Managed Object Reference [2-5](#)
 ManagedObjectReference [2-4, 2-5, 2-14](#)
 ManagedObjectReferences [2-5](#)
 Managed State Change [2-35, 2-36](#)
 Managed Sub State [2-12](#)
 manageResources [2-15](#)
 maxResults [2-26](#)
 Message [1-6, 2-26, 2-34, 2-35, 2-36](#)
 message [1-6](#)

N

Name [2-12, 2-14, 2-25](#)
 named properties [2-5](#)
 Network [2-9](#)
 networking devices [2-2](#)
 NetworkService [2-9](#)
 NewState [2-35, 2-36](#)
 nextRun [2-25](#)
 NFSServer [2-9](#)
 Notes [ii](#)
 Notification Receiver Instance [2-27](#)
 Notification Receiver Reference [2-27](#)
 Notification Reply Information [2-34, 2-35, 2-36](#)
 Notifications [2-4, 2-27](#)
 Notifications Management [2-4](#)

O

objectID [2-5](#)
 objectId [2-5](#)
 Obtaining Support [iv](#)
 OldState [2-35, 2-36](#)
 Online Programmer's Reference [1-5, 3-1, 3-2, 3-4](#)
 online reference material [3-1](#)
 operationState [2-19, 2-20](#)

P

params [2-25](#)
 Password [2-9, 2-27](#)
 password [2-8](#)
 percentComplete [2-25](#)
 platform-independent [1-6](#)
 Preface [i](#)
 Product Documentation [iii](#)
 publish/subscribe asynchronous messaging model [2-4](#)

R

RBAC [1-6](#)
 Read Community [2-10](#)
 recurring [2-25](#)
 Ref [2-17, 2-19](#)
 ref [2-20](#)
 reference to another DataObject [2-5](#)
 Related Documentation [iii](#)
 Remote Procedure Call [1-1](#)
 removeCredentials [2-11](#)
 Required Inputs [2-8](#)
 Resource Fault State Change Notification [2-36](#)
 ResourceInfo Properties [2-12](#)
 ResourceLabelInfo Attributes [2-14](#)
 ResourceLabelInfo Instance [2-14](#)
 Resource LifeCycle State [2-35](#)
 Resource LifeCycle State Change Notification [2-35](#)
 Resource Managed State Change Notification [2-35](#)
 Resource Management [2-2](#)
 ResourcePoolReference [2-12](#)
 Resource Reference [2-12](#)
 Resources [2-12, 2-13, 2-14, 2-15, 2-16, 2-17](#)
 Resources Attributes [2-12](#)
 Resource Type Code [2-12](#)
 ResourceTypeCode [2-12](#)
 Role Based Access control [2-6](#)
 role based access control [1-5](#)
 RPC [1-1](#)
 runEndTimeFrom [2-26](#)
 runEndTimeTo [2-26](#)
 runRef [2-26](#)
 runref [2-25](#)
 runStartTimeFrom [2-26](#)
 runStartTimeTo [2-26](#)
 runStatusCode [2-26](#)
 run-vframe-api.bat [3-2](#)
 run-vframe-api.sh [3-2](#)

S

sample client [3-1](#)
 Sample Client Utilities [3-2](#)
 Sample SOAP Traffic [1-3, 1-4](#)
 Sample VFrame API client code [3-1](#)
 SANManager [2-9](#)
 saveCredentials [2-11](#)
 saveResourcePools [2-16](#)
 schema definition [2-5](#)
 Security Guidelines [iv](#)
 Security Mechanism [1-5](#)
 selectionState [2-24](#)
 serialization/encoding rules [1-2](#)
 Server [2-9](#)
 Service Design [2-19, 2-20](#)
 ServiceElementEventInfo Instance [2-23, 2-24](#)
 ServiceElementInfo Instance [2-20](#)
 Service Elements [2-19](#)
 serviceName [2-19, 2-20](#)
 Service Network [2-19](#)
 service network [2-3](#)
 Service Network Design [2-3](#)
 ServiceNetworkInfo Instance [2-19](#)
 Service network templates [2-3](#)
 Service Operations [2-3, 2-21, 2-22, 2-23, 2-24](#)
 services [1-5](#)
 service template [2-3](#)
 ServiceTemplateInfo Instance [2-17](#)
 ServiceTemplateInfo instances [2-5](#)
 Service Template Management [2-3](#)
 ServiceTemplateReference [2-5](#)
 Service Templates [2-17, 2-18](#)
 Session Management [1-6](#)
 sets of endpoints operating on messages [1-5](#)
 severity [2-26](#)
 simple data-type valu [2-5](#)
 SNMP [2-2, 2-9](#)
 SOAP [1-1](#)

SOAP, WSDL, and UDDI [1-5](#)
 soapAction [1-2](#)
 SOAPAction HTTP Header [1-2](#)
 SOAP Binding [1-1](#)
 SOAP Exceptions [1-6](#)
 SOAP fault [2-6](#)
 SOAP fault element [1-4](#)
 SOAP Fault Response [1-4](#)
 SOAP fault response [1-5](#)
 SOAP Faults [1-6](#)
 SOAP Header [1-2](#)
 SOAP Request [1-3](#)
 SOAP request [1-3](#)
 SOAP Response [1-4](#)
 SOAP response [1-4](#)
 SOAP server [1-3](#)
 starting and stopping service networks [2-3](#)
 startServiceNetwork [2-21](#)
 startTime [2-25](#)
 State [2-12](#)
 status [2-25, 2-26](#)
 stopping service networks [2-3](#)
 stopServiceNetwork [2-21](#)
 Storage [2-9](#)
 stored on client [2-5](#)
 subElements [2-20](#)
 Synchronous Functionality [1-6](#)
 System Generated Flag [2-12](#)

T

TargetType [2-34, 2-35, 2-36](#)
 templateName [2-5, 2-17, 2-19, 2-20](#)
 templates [2-3](#)
 templateState [2-17](#)
 timestamp [2-26](#)
 Topic [2-27](#)
 transport [1-2](#)
 Transport Protocols [1-2](#)

type [2-5, 2-25](#)
 typeCode [2-26](#)

U

UNIX/LINUX [3-2](#)
 unManageResources [2-15](#)
 Unpacking the API/SDK Files [3-1](#)
 unzip utility [3-1](#)
 URI [1-2, 2-27](#)
 URL [1-6](#)
 Username [2-9, 2-27](#)
 username [2-8](#)
 utf-8 [1-2](#)
 UTF-8 encoding [1-2](#)

V

value [2-5](#)
 verifyServiceNetwork [2-21](#)
 Version [2-27](#)
 version [2-2](#)
 VFAuthorizationFault [2-6](#)
 VFFault [2-6](#)
 VFInvalidValueFault [2-7](#)
 VFrame API [1-1](#)
 VFrame API SDK [3-1](#)
 VFrame APIs WSDL file [1-2](#)
 VFrame Data Center [1-1](#)
 VFrame Data Center API SDK [3-1](#)
 VFrame Data Center SDK [3-1](#)
 VFrame Directory Contents [3-2](#)

W

Web Services Description Language [1-1](#)
 WinZip [3-1](#)
 working with service networks [2-3](#)

Write Community [2-10](#)
WSDL [1-1](#)
WSDL and SOAP [1-1](#)
WSDL code [3-4](#)
WSDL Documentation [3-1](#)
WSDL files [1-5, 3-1](#)
WSDL Files Location [1-5](#)
WSDL SOAP binding [1-1](#)
WS-I [1-5](#)
WS-I compliant [1-2](#)

X

XML schema [2-6](#)
XML Schema Part 2 [1-2](#)
XSD [3-4](#)