VISTALINK

SYSTEM MANAGEMENT GUIDE

Version 1.6

December 2010

Department of Veterans Affairs
Office of Information and Technology
Product Development

Revision History

Date	Description	Author
12/03/10	VistALink Version 1.6 release.	Product Development Services Security Program VistALink development team. Albany, NY OIFO: • Developer—Mike Kilmade • Developer—Liz Defibaugh Bay Pines, FL OIFO: • Development Manager—Charles Swartz • Tester—Jason Woodyard Oakland, CA OIFO: • Developer—Kyle Clarke • SQA—Gurbir Singh • Technical Writer—Susan Strack
05/2006	Initial VistALink Version 1.5 release.	Jim Alexander, technical writer Dawn Clark, project manager

Table i. Revision History

Contents

Figures

Tables

Orientation

This System Management Guide is intended for use in conjunction with the VistALink software. It outlines the details of VistALink-related software and gives guidelines on how the software is used within Health*e*Vet-Veterans Health Information Systems and Technology Architecture (VistA).

The intended audience of this manual is all key stakeholders. The primary stakeholder is the Product Development Security Program team. Additional stakeholders include:

- Health*e*Vet-VistA application developers of Web-based applications in the WebLogic 9.2 and 10.x Application Server environment.
- Information Resource Management (IRM) and Information Security Officers (ISOs) at Veterans Affairs Medical Centers (VAMCs) responsible for computer management and system security.
- Product Support (PS).
- VA facility personnel who will be using Health*e*Vet-VistA Web-based applications running in the WebLogic 9.2 and 10.x Application Server environment.

Document Overview

This manual provides information on the management of VistALink resource adapters and servers. It contains detailed information on:

- Deploying VistALink on Java 2 Enterprise Edition (J2EE) Servers
- J2EE Logging
- VistALink's Institution Mapping
- The VistALink administration console
- Monitoring Adapters
- M listener management
- VistALink security
- Troubleshooting

Its intended audience includes server administrators and IRM Information Technology (IT) specialists at VA facilities, as well as developers of Java applications requiring communication with VistA/M.

Generally, the installation and maintenance instructions presented here assume the use of Windows as the client operating system. Where appropriate, separate steps are displayed for Linux.

Terminology

The term *resource adapter* is often shortened in this guide to "adapter," and is also used interchangeably with the term *connector*.

Text Conventions

File names and directory names are set off from other text using bold font (e.g., **config.xml**). Bold is also used to indicate GUI elements, such as tab, field, and button names (e.g., "press **Delete**").

All caps are used to indicate M routine and option names (e.g., XMINET). All caps used inside angle brackets indicate file names to be supplied by the user. Example:

```
<JAVA_HOME>\bin\java -Dlog4j.configuration=file:///c:/localConfigs/mylog4j.xml
```

Names for Java objects, methods, and variables are indicated by Courier font. Snapshots of computer displays also appear in Courier, surrounded by a border:

```
Select Installation Option: LOAD a Distribution
Enter a Host File: XOB_1_6.KID
```

In these examples, the response that the user enters at a prompt appears in bold font:

```
Enter the Device you want to print the Install messages.
You can queue the install by enter a 'Q' at the device prompt.
Enter a '^' to abort the install.

DEVICE: HOME// <Enter> TELNET PORT
```

Boldface text is also used in code and deployment descriptor samples to indicate lines of particular interest, discussed in the preceding text:

```
<?xml version="1.0"?>
<weblogic-connector xmlns="http://www.bea.com/ns/weblogic/90"
xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
xsi:schemaLocation="http://www.bea.com/ns/weblogic/90
http://www.bea.com/ns/weblogic/90/weblogic-ra.xsd">

   <!-- For new ADAPTER-level jndi-name, recommend using value of
connection instance JNDI name, appended with Adapter -->
   <jndi-name>vlj/testconnectorAdapter</jndi-name>

   <enable-global-access-to-classes>true</enable-global-access-to-classes>
```

The following symbols appear throughout the documentation to alert the reader to special information or conditions.

Symbol	**Description**
ℹ	Used to inform the reader of general information and references to additional reading material, including online information.
⚠	**Used to caution the reader to take special notice of critical information.**

Additional Resources

VistALink Web Site

The VistALink website summarizes VistALink architecture and functionality and presents status updates:

http://vaww.vista.med.va.gov/vistalink/.

VistALink Documentation Set

The following is the VistALink 1.6 end-user documentation set:

- *VistALink 1.6 Installation Guide*: Provides detailed instructions for setting up, installing, and configuring the VistALink listener on VistA/M servers and the VistALink resource adapter on J2EE application servers. Its intended audience includes server administrators, IRM IT specialists, and Java application developers.
- *VistALink 1.6 System Management Guide* (this manual): Contains detailed information on system management for VistALink adapters and M listeners.
- *VistALink 1.6 Developer Guide*: Contains detailed information about workstation setup, re-authentication, institution mapping, executing requests, VistALink exceptions, Foundations Library utilities, and other topics pertaining to writing code that uses VistALink.
- *VistALink 1.6 Release Notes*: Lists all new features included in the release.

VistALink end-user documentation can be downloaded from the VA Software Document Library (VDL) Web site:

http://www.va.gov/vdl/application.asp?appid=163

VistALink end-user documentation and software can be downloaded from any of the **anonymous.software** directories on the Office of Information Field Office (OIFO) File Transfer Protocol (FTP) download sites::

• Albany OIFO	ftp.fo-albany.med.va.gov
• Hines OIFO	ftp.fo-hines.med.va.gov
• Salt Lake City OIFO	ftp.fo-slc.med.va.gov
• Preferred Method	download.vista.med.va.gov

The documentation is made available online in Microsoft Word format and Adobe Acrobat Portable Document Format (PDF). The PDF documents *must* be read using the Adobe Acrobat Reader (i.e., ACROREAD.EXE), which is freely distributed by Adobe Systems Incorporated at the following Web address:

http://www.adobe.com/

WebLogic Systems

At the current time, VistALink 1.6 has been tested and is supported on WebLogic Server 9.2 and 10.x, only. WebLogic product documentation can be found at the following website:

http://edocs.bea.com/.

DISCLAIMER: The appearance of any external hyperlink references in this manual does not constitute endorsement by the Department of Veterans Affairs (VA) of this Web site or the information, products, or services contained therein. The VA does not exercise any editorial control over the information you may find at these locations. Such links are provided and are consistent with the stated purpose of this VA Intranet Service.

1. Introduction

1.1. VistALink 1.6 Overview

The VistALink resource adapter is a transport layer that provides communication between Health*e*Vet-VistA Java applications and VistA/M servers, in both client-server and n-tier environments. It allows Java applications to execute remote procedure calls (RPCs) on the VistA/M system and retrieve results, synchronously. VistALink is also referred to as "VistALink J2M."

VistALink 1.6 consists of Java-side adapter libraries and an M-side listener:

- The adapter libraries use the J2EE Connector Architecture (J2CA 1.5) specification to integrate Java applications with legacy systems.
- The M listener process receives and processes requests from client applications.

1.2. WebLogic Updates Project

In support of the Department of Veterans Affairs Information Technology application Modernization effort, the three applications VistALink, Fat-client Kernel Authentication and Authorization (FatKAAT), and Kernel Authentication and Authorization for the Java 2 Enterprise Edition (KAAJEE) have been developed. Based on the direction of the Technical Review Model (TRM) and in order to support applications that upgrade to the new WebLogic Server versions 9.2 and 10.x, this project is required. The scope of the project is to upgrade these three applications to work with the WebLogic Server versions 9.2 and 10.x.

2. Deploying VistALink Adapters on J2EE

2.1. RAR Overview

On J2EE systems, J2CA-compliant adapters, such as VistALink, are deployed in resource adapter archive (RAR) files. RAR files are analogous to Enterprise Archive (EAR) files and Web Archive (WAR) files, except that they are exclusively for resource adapters (J2CA connectors).

NOTE: Oracle (formerly known as BEA) recommends that RARs be deployed in exploded format.

Deployment characteristics of VistALink RARs running in WebLogic domains:

- VistALink adapters are meant to be deployed as standalone J2CA adapters, e.g., independent of individual J2EE applications.
- For a J2EE application to use a VistALink adapter, the adapter must be running in the same JVM as the J2EE application.
- If a J2EE application using VistALink is targeted to multiple WebLogic servers, so must the VistALink adapter(s) it is using.
- A single VistALink adapter deployment can target multiple WebLogic servers in a domain.
- A VistALink adapter runs individually on each server it is targeted to; it is not aware of itself running on other servers in the domain.
- There are no cluster-aware features or configuration settings for standalone J2CA adapters such as VistALink in WebLogic. In particular, there is no:
    - failover across servers
    - load-balancing across servers

The configurable settings for deployment of any VistALink adapter are contained in the following locations:

- **weblogic-ra.xml**: contains WebLogic-specific deployment descriptor configuration settings (such as initial and maximum pool sizes) and JNDI-related properties that must be modified for each adapter, to distinguish one adapter from another in JDNI.
- **VistALink configuration file**: contains VistALink-specific configuration settings, such as access and verify codes for the connector proxy user. The VistALink configuration file contains settings for *all* deployed VistALink connectors. It is in a single location on a given server, in a folder that the deployer places on the server's classpath.

NOTE: With the deployment descriptor templates provided with VistALink 1.6, in most case the **ra.xml** deployment descriptor does not need to be modified.

2.1.1. Summary of RAR Changes Since Previous Version

- **vljFoundationsLib and vljConnector jars**: New versions provide a J2CA 1.5 compliant adapter implementation.
- **Deployment Descriptor Changes:** The format of both the ra.xml and weblogic-ra.xml descriptors is different. Existing adapters' deployment descriptors need to be updated. Using the provided sample/template descriptors, only weblogic-ra.xml will need to be modified when creating a VistALink RAR:
  - **META-INF/ra.xml**: The template version provided in the VistALink distribution is updated to be J2CA 1.5 compliant. In most cases, it no longer needs any further editing when creating a RAR.
  - **META-INF/weblogic-ra.xml**: The template version provided in the VistALink distribution is updated to be J2CA 1.5 compliant. It must be edited for each RAR deployed.
  - **META-INF/MANIFEST.MF**: The template version provided in the VistALink distribution is updated to support use of J2EE Shared Libraries by RARs.
- **lib/saxpath.jar, lib/jaxen-core.jar, and lib/jaxen-dom.jar** are no longer needed in the RAR as their functionality has been replaced by the XPath implementation introduced in Java 5.
- **lib/xbean.jar** no longer needed in the RAR since an implementation of XML Beans is included in WebLogic 9/10
- **Linked adapters** (link-ref mechanism) no longer supported for VistALink 1.6 adapters
- **Development System Classloading:** Automatic classloading of all jars in RAR is now supported by WebLogic 9.x/10. Library jars included in RAR no longer need to be manually added to the server classpath.
- **Production System Classloading:** Oracle recommends removing all jars from RARs on production systems, and deploying them as J2EE shared libraries instead (to reduce resource consumption – replacement for resource-saving aspect of the 8.1 link-ref mechanism). When deployed as J2EE shared libraries, the jars supporting VistALink RARs no longer need to be manually added to the server classpath.

NOTE: VistaLink 1.6 is upgraded to be a J2CA 1.5-compliant adapter, and runs in WebLogic 9.x and 10. The previous version of VistALink (1.5) implemented the J2CA 1.0 standard, and runs in WebLogic 8.1.

2.2. Creating a RAR

2.2.1. Exploded 1.6 RAR Layout, Production Systems

The recommended contents of an exploded VistALink RAR folder, for production systems, are:

(root)	empty
META-INF\	- MANIFEST.MF (contains information needed to use J2EE Shared Libraries) - ra.xml (J2EE standard Deployment Descriptor) - weblogic-ra.xml (WLS-specific Deployment Descriptor)

The app-J2EE\Rar-Prod-Template directory in the VistALink distribution zip provides an exploded RAR in this structure.

NOTE: On production systems, Oracle recommends that the supporting libraries (JAR files) be deployed as J2EE shared libraries to minimize the consumption of server resources (hence the recommended exploded RAR for a production system contains only deployment descriptors).

2.2.2. Exploded 1.6 RAR Layout, Non-Production Systems

The recommended contents of an exploded VistALink RAR folder, for **non-**production systems, are:

(root)	Main VistALink libraries: - vljConnector-1.6.0.jar - vljFoundationsLib-1.6.0.jar
lib\	Supporting libraries: - log4j-1.2.13.jar
META-INF\	- MANIFEST.MF (a required RAR artifact) - ra.xml (J2EE standard Deployment Descriptor) - weblogic-ra.xml (WLS-specific Deployment Descriptor)

The app-J2EE\Rar-Dev-Template directory in the VistALink distribution zip provides an exploded RAR in this structure.

NOTE: Libraries contained in RARs are automatically loaded onto a high-level classloaders, and are thus available to other deployed applications (EARS, WARs, etc) running in the same Java Virtual Machine (JVM).

2.2.3. 1.6 Deployment Descriptor: ra.xml

A template **ra.xml** version is provided in the VistALink distribution zip file, in each of the example rar folders. It has been updated to be compatible with 1.5 of the J2CA specification.

In most cases, you can copy this template ra.xml deployment descriptor as-is from the VistALink example RAR for all your RARs, with no further need to edit it.

```
<?xml version="1.0" encoding="UTF-8"?>
<connector xmlns="http://java.sun.com/xml/ns/j2ee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/connector_1_5.xsd" version="1.5">
 <description>J2M VistALink adapter that allows RPC execution from Java/J2EE to
  M/VistA</description>
 <display-name>VistaLinkAdapter</display-name>
 <vendor-name>Foundations</vendor-name>
 <eis-type>VistA</eis-type>
 <resourceadapter-version>1.6</resourceadapter-version>

 <!-- NOTE: In VistALink v1.6, deployed in WebLogic, in most cases this
    VistALink-distributed ra.xml file does NOT require any edits/changes. Instead,
    all edits/changes should be made in weblogic-ra.xml. -->

 <resourceadapter>
  <resourceadapter-
class>gov.va.med.vistalink.adapter.spi.VistaLinkResourceAdapter</resourceadapter-
class>
  <outbound-resourceadapter>
   <connection-definition>
    <managedconnectionfactory-
class>gov.va.med.vistalink.adapter.spi.VistaLinkManagedConnectionFactory</managedcon
nectionfactory-class>
    <config-property>
     <description>placeholder property for connectorJndiName whose presence allows
overriding in weblogic-ra.xml</description>
     <!-- NOTE: On WebLogic, with VistALink v1.6, DON'T edit the ra.xml
      connectorJndiName value. INSTEAD, edit the weblogic-ra.xml connection instance
      connectorJndiName property. It overrides this one! -->
     <config-property-name>connectorJndiName</config-property-name>
     <config-property-type>java.lang.String</config-property-type>
     <config-property-value>vljPlaceholder</config-property-value>
    </config-property>
    <connectionfactory-
interface>javax.resource.cci.ConnectionFactory</connectionfactory-interface>
    <connectionfactory-impl-
class>gov.va.med.vistalink.adapter.cci.VistaLinkConnectionFactory</connectionfactory
-impl-class>
    <connection-interface>javax.resource.cci.Connection</connection-interface>
    <connection-impl-
class>gov.va.med.vistalink.adapter.cci.VistaLinkConnection</connection-impl-class>
   </connection-definition>
   <transaction-support>NoTransaction</transaction-support>
   <reauthentication-support>false</reauthentication-support>
  </outbound-resourceadapter>
 </resourceadapter>
</connector>
```

Figure 2-1. 1.6 Deployment Descriptor Template (ra.xml)

2.2.4. 1.6 Deployment Descriptor: weblogic ra.xml

The location of the provided, template **weblogic-ra.xml** deployment descriptor is also the META-INF subfolder inside the adapter RAR file (packaged or exploded). The VistALink zip distribution provides example adapters with pre-configured **ra.xml** and **weblogic-ra.xml** files.

Unlike the **ra.xml** deployment descriptor, you will need to edit the **weblogic-ra.xml** descriptor with unique values for each RAR you deploy. The key properties to modify are:

Property	Value
`jndi-name` (top/adapter-level)	Name should be unique across all adapters. This is a NEW property since the previous version of VistALink. Needs to be DIFFERENT than the value of jndi-name used for the connection instance. Recommend setting to the value of the connection-instance jndi-name, appended with "Adapter", e.g., "*vlj/SiouxFalls438Adapter*".
`jndi-name` (connection-instance)	Name should be unique across all adapters. The value of this property is the JNDI lookup string that applications will use to retrieve the adapter, e.g., "*vlj/SiouxFalls438*". Recommend the JNDI name be alphanumeric, also OK to include the characters: - _ / ( ) [ ]
`connectorJndiName` (value)	Set to the same value as the connection-instance jndi-name.

Example:

```
<?xml version="1.0"?>
<weblogic-connector xmlns="http://www.bea.com/ns/weblogic/90"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.bea.com/ns/weblogic/90
http://www.bea.com/ns/weblogic/90/weblogic-ra.xsd">

  <!-- Warning: The order the elements appear in complex elements is usually
important. It is a good idea to validate and test the weblogic-ra.xml document
before committing. -->

  <!-- For new ADAPTER-level jndi-name, recommend using value of connection
instance JNDI name, appended with "Adapter" -->
  <jndi-name>vlj/SiouxFalls438Adapter</jndi-name>

  <enable-global-access-to-classes>true</enable-global-access-to-classes>

  <outbound-resource-adapter>
    <connection-definition-group>
      <connection-factory-
interface>javax.resource.cci.ConnectionFactory</connection-factory-interface>

      <default-connection-properties>
        <pool-params>
          <initial-capacity>1</initial-capacity>
          <max-capacity>5</max-capacity>
          <capacity-increment>1</capacity-increment>
          <shrinking-enabled>true</shrinking-enabled>
          <shrink-frequency-seconds>1800</shrink-frequency-seconds>
          <highest-num-waiters>2147483647</highest-num-waiters>
          <connection-creation-retry-frequency-seconds>30</connection-creation-
retry-frequency-seconds>
          <connection-reserve-timeout-seconds>0</connection-reserve-timeout-
seconds>
          <test-frequency-seconds>3600</test-frequency-seconds>
          <profile-harvest-frequency-seconds>30</profile-harvest-frequency-seconds>
          <ignore-in-use-connections-enabled>false</ignore-in-use-connections-
enabled>
          <match-connections-supported>true</match-connections-supported>
        </pool-params>
        <transaction-support>NoTransaction</transaction-support>
        <reauthentication-support>false</reauthentication-support>
      </default-connection-properties>

      <connection-instance>
        <description>This is the connection and JNDI name that applications
          will be accessing.</description>
        <jndi-name>vlj/SiouxFalls438</jndi-name>
        <connection-properties>
          <properties>
            <property>
              <!-- connectorJndiName value should be the same value as
                   connection instance jndi-name a few lines above -->
              <name>connectorJndiName</name>
              <value>vlj/SiouxFalls438</value>
            </property>
          </properties>
        </connection-properties>
      </connection-instance>

    </connection-definition-group>
  </outbound-resource-adapter>
</weblogic-connector>
```

Figure 2-2. 1.6 Deployment Descriptor Template (weblogic ra.xml)

2.2.5. Steps to Create a 1.6 RAR

1. Create a folder, in a file location accessible on the admin server, for the new RAR. Name the folder in a way that easily identifies the adapter, since the folder name will become the deployed adapter name in WebLogic.
2. Copy the files needed for the new RAR from the VistALink zip distribution to the new RAR folder:
   - **Production Systems**: Copy the contents of the app-J2EE\Rar-Prod-Template folder from the VistALink zip distribution to the new RAR folder.
   - **Non-production systems**: Copy the contents of the app-J2EE\Rar-Dev-Template folder from the VistALink zip distribution to the new RAR folder.
3. Edit the **weblogic-ra.xml** deployment descriptor in the new RAR, to have the appropriate JNDI names.

2.3. VistALink Connector Configuration File

VistALink-specific resource adapter (connector) settings are stored in a separate, host-file-system-based connector configuration file, named **gov.va.med.vistalink.connectorConfig.xml**. You can edit this file manually or using the Configuration Editor (see the section "Configuration Editor").

The rules for the VistALink configuration file are:

- The file must be named "gov.va.med.vistalink.connectorConfig.xml".
- The file must be placed in a folder that is on the 'java' classpath of the JVM of each WebLogic server instance deploying VistALink connectors.
- Because the **gov.va.med.vistalink.connectorConfig.xml** file holds login credentials for accessing VA VistA systems, it must be protected. On Linux systems, access to the folder containing the file should be restricted to the account or group under which WebLogic runs. Access to the host file system should be protected on all J2EE systems.

When the server deploys a VistALink connector, the connector does the following:

1. Gets the `connectorJndiName` property value from **weblogic-ra.xml** deployment descriptor.
2. Loads **gov.va.med.vistalink.connectorConfig.xml**, via the server 'java' classpath.
3. Looks in the **VistALink** configuration file for a `<connector>` entry with a matching **jndiName** attribute.
4. Uses the settings from the matching entry (`ip, port, access-code, verify-code,` etc.) to configure the connector.

NOTE: Although you can manually edit the VistALink configuration file, a Configuration Editor UI is provided as part of the VistALink console (see the "Configuration Editor." section in the *VistaLink Administration Console* chapter.)

2.3.1. Connector Entry Format

The VistALink configuration file should contain one `<connector>` entry per connector module/adapter that you will deploy to your J2EE server. Each entry should have a unique `jndiName`. Each `<connector>` entry describes the characteristics of the M listener that a particular connector module/adapter will connect to.

The following example shows the format of the VistALink configuration file (with a configuration of a single listener):

```
<?xml version="1.0" encoding="UTF-8"?>

<connectors encryptionScoped="false"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="connectorConfig.xsd"
xmlns="http://med.va.gov/vistalink/adapter/config"
>
   <connector
      jndiName="vlj/SiouxFalls438"
      primaryStation="438"
      ip="test.somewhere.va.gov"
      port="8000"
      access-code="joe.123"
      verify-code="ebony.432"
      timeout="30"
      always-use-default-as-min="false"
      enabled="true"
   />

</connectors>
```

Figure 2-3. VistALink Configuration File Format Example

2.3.2. Connector Settings

The configuration file should contain one or more <connector> elements (entries) inside the top-level <connectors> element.

The basic settings for each `<connector>` entry are as follows:

- **`jndiName`** (required**)**: Uniquely identifies each entry. Should be the same JNDI name as you will specify in the **weblogic-ra.xml** descriptor for your connector. This setting is also used to create the JNDI half of the mappings between station numbers and connector JNDI names.

  NOTE: The value must start with an alphanumeric character, and following that can contain only alphanumeric plus the following punctuation characters: - _ / ( ) [ ]

- **`primaryStation`** (required): Station number of the M/Kernel system connected to. VistALink's institution mapping functionality maps this station number to the connector's JNDI name, so the application can retrieve the connector by the station number. This entry is checked against the DEFAULT INSTITUTION field (#217) of the KERNEL SYSTEM PARAMETERS file (#8989.3) at runtime, when connections are made.

- **`ip`** (required): IP address of the M VistALink listener to connect to (either numeric or mnemonic).
- **`port`** (required): Port of the M VistALink listener to connect to.
- **`access-code`** and **`verify-code`** (required): The access and verify code credentials for the connector proxy user to connect to the M VistALink listener.

  NOTE: When entering access and verify codes directly in the config file (not using the configuration editor), if the codes contain the following special characters, they need to be entered as follows:

special char	enter as
<	`&lt;`
&	`&amp;`
"	`&quot;`
'	`&apos;`

  or alternatively, use the configuration editor.

- **`encrypted`** (optional)**:** true | false. When the access/verify codes are encrypted by VistALink, this attribute is set to true. If you need to manually (outside the configuration editor) update the access / verify code, however, set "encrypted" to false so that VistALink will know the access/verify code is not encrypted.
- **`enabled`** (required): true | false. If false, the connector will not deploy. Use this attribute primarily to retain inactive configurations in your configuration file.
- **`timeout`** (optional): Sets the default span of time in seconds socket will wait for response from M (e.g., waiting for an RPC to execute) and after which connection is automatically terminated.

2.3.3. Advanced Connector Settings

- **`always-use-default-as-min`:** If true, the default timeout will be the lowest timeout value allowed for this connector; it will override any programmatically specified lower value set by an application.
- **`timestamp`:** When a connector entry is edited, the configuration editor automatically stamps the entry with a timestamp.

2.3.4. File-Wide Setting: encryptionScoped

- **`encryptionScoped`:** This attribute should not be edited directly, and instead should be modified only when using the VistALink administration console to modify the encryption type for all configuration file entries. The file-wide <connectors> attribute "encryptionScoped" reflects whether entries have been encrypted with domain-scoped encryption or not. If domain scoped, the domain name is included in the encryption, making it difficult to use encrypted values on other WebLogic domains.

2.3.5. Obsolete Settings

- **`primaryStationSuffix`:** This attribute has been eliminated. Any primary station numbers requiring an alpha suffix, should instead be entered as part of the "primaryStation" attribute, i.e., primaryStation="200M". Note: If VA institution rules are being used, only 200-series (Austin Information Technology Center) station numbers can have alpha suffixes for the primary station number.

2.4. Deploying J2EE Shared Libraries for VistALink (Production Systems Only)

REF: To minimize the consumption of server resources, Oracle recommends that supporting libraries for RARs should be deployed as J2EE Shared Libraries, on production systems.

On production systems only, prior to deploying any VistALink RARs, you should install the following jars as J2EE shared libraries:

Jar Name	J2EE Shared Library Deployment Name
vljConnector-1.6.0.jar	vljConnector(1.6,1.6)
vljFoundationsLib-1.6.0.jar	vljFoundationsLib(1.6,1.6)
log4j-1.2.13.jar	log4j-1

To deploy a jar as a shared library (Production Systems):

1. Place the jar in a folder location accessible on the admin server.
2. Log on to the WebLogic console.
3. Select Lock & Edit so you can modify the domain configuration.
4. Navigate to the Deployments page.
5. Click Install. Navigate to the jar, select it and click Next.
6. Choose the server(s) to target. Choose all servers that you will be deploying VistALink RARs on, and click Next.
7. Leave the Name value as-is, and click Finish.
8. Click Activate Changes to complete the deployment.

2.5. Deploying a RAR

For each M system you want your J2EE system to connect to, you need to create and deploy a separate VistALink RAR (exploded or packaged). Each RAR contains two deployment descriptors, **ra.xml** and **weblogic-ra.xml**. These need to be configured with the unique settings that describe how the adapter should be deployed on the J2EE system and how it should connect to a specific M system.

To deploy a new VistALink adapter:

1. If you are deploying to a production system, ensure that you have deployed the necessary supporting libraries for VistALink as J2EE Shared Libraries before proceeding any further.
2. Create the RAR (see *Creating a RAR* section above).
3. Add an entry in the **gov.va.med.vistalink.connectorConfig.xml** file for the new adapter, on all servers that the connector will be deployed on. Its jndiName attribute should match the values used in weblogic-ra.xml in the <connection-instance> section, for <jndi-name> and connectorJndiName (see *VistALink Connector Configuration File* section above).
4. Propagate the updated gov.va.med.vistalink.connectorConfig.xml file to all servers in the domain hosting VistALink adapaters.
5. Use WebLogic to target and deploy the new RAR to one or more servers.
6. Verify that the new adapter is working correctly on each targeted server, using the VistALink administration console (see *The VistALink Administration Console* for more information).

   Even though the adapter may deploy successfully from WebLogic's point of view, the test for configuration completion is whether the adapter can successfully connect to the M system.

2.6. Updating Already-Deployed Adapters

If you update an adapter setting in the **VistALink** configuration file, **weblogic-ra.xml**, or **ra.xml**, you do not need to bounce the server to activate the changed setting. It is sufficient to *update* the adapter in the WebLogic console to make the changed setting(s) active.

NOTE: While the adapter is being updated, it may be briefly unavailable to applications.

To update an adapter:

1. Log on to the WebLogic console.
2. Select Lock & Edit so you can modify the domain configuration.
3. Navigate to the Deployments page.
4. Select the adapter to update and click Update. Complete the steps necessary to finish the update.
5. Click Activate Changes.
6. Verify that the adapter state is Active.
7. Verify that the updated adapter is working correctly using the VistALink administration console (see "*The VistALink Administration Console*" section for more information).

2.7. DNS Updates and VistALink Adapters

If the value of the "Ip" property used to configure a VistALink 1.6 adapter's destination M system is a domain name, then DNS updates to the IP address corresponding to that DNS name ***can*** be received dynamically/automatically by the adapter, with a few caveats. This is a change from previous versions of VistALink. The caveats are:

1. The JVM's "cache forever" setting may force DNS lookup caching at the JVM level. For VistALink adapters to receive DNS updates dynamically, the JVM must be configured to turn off DNS caching. Otherwise, the JVM will resolve the DNS name once, at deployment, and cache that name until the next time the JVM restarts.

   This setting is usually controlled by the "networkaddress.cache.ttl" JVM property in the java.security file, in the lib/security subdirectory of the JRE used for WebLogic. For example:

   ```
   # The Java-level namelookup cache policy for successful lookups:
   #
   # any negative value: caching forever
   # any positive value: the number of seconds to cache an address for
   # zero: do not cache
   #
   # default value is forever (FOREVER). For security reasons, this
   # caching is made forever when a security manager is set.
   #
   # NOTE: setting this to anything other than the default value can
   #       have serious security implications. Do not set it unless
   #       you are sure you are not exposed to DNS spoofing attack.
   #
   # networkaddress.cache.ttl=-1
   ```

   In order to disable JVM-level DNS caching, the "networkaddress.cache.ttl" property must be uncommented for the JRE used by WebLogic, for all servers in the domain, and set to some value of 0 or greater.

   NOTE: The "Java Home" value may need to be explicitly set in the "Configuration | Server Start" page for each server in the domain (use the WebLogic console), in order for the change in java.security to be picked up.

   NOTE: Assuming servers running VistALink are running inside a secure intranet, the security risk of disabling DNS caching is low (if an organization is using DNS at all to dynamically update internal IP addresses, then it has presumably already evaluated the risk to be acceptable.)

2. Existing connections in the connection pool remain connected to previous destination. If the DNS is updated, existing open connections to the previous DNS-resolved system will remain until those connections close. The M-side "J2EE Connection Timeout" setting in the M-side VistALink site parameters file determines life of inactive connections. Since the current recommended value is 1 day, even inactive connections could stay open to the former destination for quite some time, and if connections are used, they could stay open even longer.

Some strategies to ensure existing connections are terminated when DNS is changed for the M system include:

- If the M system at the old address is shut down, all open connections are terminated automatically.
- If the M system manager kills all existing VistALink connections, this will remove any old connections from pool (if won't disrupt service to remaining VistALink clients, if any).
- On the J2EE side, if the J2EE system manager 1) stops the adapter, and 2) updates the adapter, this will shut down the pool and all old connections.

2.8. Pool Size Management/Tuning

Pool size management is a key factor in enhancing VistALink's performance on a J2EE server (and minimizing its impact on M servers). It is costly to have either too many unused open connections or not enough for incoming requests. Connection pool sizing will be part of the art of system tuning in a HealtheVet-VistA environment.

On the WebLogic server, you can use the WebLogic console and the **weblogic-ra.xml** deployment descriptor to control the characteristics of the connection pool for each deployed resource adapter.

Key settings include:

- **`Initial Capacity`:** The number of connections to create for the connection pool. Pool creation happens on initial deployment and on server startup. Higher numbers for this setting can add additional time to the server startup process.
- **`Max Capacity`:** The high point of expansion for the connection pool. You may want to set this based on the highest load you can place on the M system you are connected to (potential work as well as license slot usage). On the J2EE side, if all connections are in use and the Max Capacity setting is reached, applications requesting a connection will be thrown a `ResourceException`.
- **`Shrinking Enabled and Shrink Frequency Seconds`:** This allows pools to shrink when the number of requests has slowed down and created connections are inactive for a given set of time. Enabling shrinking is recommended in most cases, to reduce the number of inactive connections using license slots on M systems.

3. Configuring log4j Logging

The VistALink Java code base has been instrumented with log4j logging statements. **log4j** is an open-source logging package distributed under the Apache Software license.

It can be helpful in debugging and troubleshooting to review the output information contained in the log files produced at runtime. System administrators can configure log4j to log either VistALink errors only or VistALink errors with debug information.

●	**REF:** Learn more about the log4j tool at: http://logging.apache.org/log4j/docs/ .

3.1. log4j Configuration Overview

A log4j configuration file contains:

- **appender entry(s):** Configuration entries describing the destinations for logging information
- **logger entry(s):** Configuration entries describing the level specific information will be logged. Loggers are typically organized by Java package and class name.

The configuration file is used by log4j to set up a JVM-wide logging configuration. Because this configuration spans applications, Health*e*Vet-VistA applications do not control logging configurations individually. Instead, a single log4j configuration file must be set up with the logging configuration for all Health*e*Vet-VistA applications running on a particular JVM.

3.2. Configuring log4J

Currently, we recommend the following steps for configuring log4j:

1. Create a single file named "log4j.xml" to hold the loggers and appenders for all Health*e*Vet-VistA applications on your J2EE system.
2. Add in all the appender elements required to set up as many logging destinations as you need for your system.
3. Add in all the logger elements required to configure logging for your Health*e*Vet-VistA applications running on your server. (VistALink provides a list of available loggers, which you can use to configure logging coming from VistALink classes.)
4. Place this file in a Health*e*Vet-VistA configuration folder on your J2EE server JVM classpaths.

   The purpose of the folder is to hold configuration files for all Health*e*Vet-VistA applications, including VistALink.

If you perform steps 1-4 above, log4j will find the **log4j.xml** file on your server classpaths, and will be able to establish a JVM-wide log4j logging configuration for all applications running in the JVM.

3.2.1. Alternative Approach

Alternatively, you can place a log4j configuration file on the file system and pass its name and location as the `log4j.configuration` system property to the JVM at startup. This file does not need to be on the JVM classpath.

The following example shows how to start a JVM with a log4j configuration stored in a non-log4j.xml file, in a folder that is not on the server classpath:

```
<JAVA_HOME>\bin\java -Dlog4j.configuration=file:///c:/localConfigs/mylog4j.xml
```

3.3. Sample log4j Configuration Files

To create the log4j configuration file, the system administrator can do *one* of the following:

- Use the information above to add to an overall log4j configuration file
- Copy one of the sample log4j configuration files provided in the VistALink distribution zip file or
- Just copy some of the logger elements in the sample files.

The sample log4j configuration files can be located in the log4j directory inside the VistALink distribution zip file. There are two sample log4j configuration files:

File Name	Description
log4jSampleJ2EEConfig.xml	Example configuration for a J2EE system, turning DEBUG-level logging on for VistALink classes of interest on a J2EE system. **Note:** Turning on 'debug' level can adversely affect system performance.
log4jSampleJ2SEConfig.xml	Example configuration for a J2SE (Java client-M server) application, turning DEBUG-level logging on for VistALink classes of interest for client/server applications.

Figure 3-1. Sample log4j configuration files

The following is a simple log file example with one appender element, two logger elements and a root logger, reduced from "log4jSampleJ2EEConfig.xml":

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">

<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">

  <!-- APPENDERS -->
  <appender name="verboseDailyRollingFileAppender"
class="org.apache.log4j.DailyRollingFileAppender">
    <param name="File" value="/YOURLOGDIR/vlj.log"/>
    <param name="DatePattern" value="'.'yyyy-MM-dd"/>
    <layout class="org.apache.log4j.PatternLayout">
     <param name="ConversionPattern" value="%-4r %d{ISO8601} [%t] %-5p %C:%M:%L -
%m%n"/>
    </layout>
  </appender>

  <!-- LOGGERS -->
  <!-- DEBUG-level loggers for VistaLink packages, override ROOT logger level
       in production, usually turn on DEBUG level logging for troubleshooting
specific problems only -->

  <logger name="gov.va.med.vistalink.adapter.spi.VistaLinkManagedConnectionFactory"
additivity="false" >
        <level value="debug" />
     <appender-ref ref="verboseDailyRollingFileAppender"/>
  </logger>

  <logger name="gov.va.med.vistalink.adapter.spi.VistaLinkManagedConnection"
additivity="false" >
        <level value="debug" />
     <appender-ref ref="verboseDailyRollingFileAppender"/>
  </logger>

  <!-- ROOT level logger: usually good to log all ERROR-level entries regardless of
source -->
  <root>
        <level value="error" />
     <appender-ref ref="verboseDailyRollingFileAppender"/>
  </root>

</log4j:configuration>
```

Figure 3-2. Log4j configuration file example

3.4. VistALink Core log4j Categories

VistALink core log4j categories are documented in the spreadsheet provided in the **log4j** subdirectory of the VistALink distribution file.

To obtain a full log4j logger category name for a given package and class, concatenate the package with the class. For example, the logger for: `gov.va.med.vistalink.adapter.spi` (package) and `VistaLinkManagedConnection` (class) is `gov.va.med.vistalink.adapter.spi.VistaLinkManagedConnection`.

Each logger category provides a description and notes the supported logger levels (e.g., DEBUG and ERROR). You can then use the logger category name(s) to set up loggers in your log4j configuration files to log information from specific parts of the VistALink package, as needed.

4. Institution Mapping

Applications using VistALink need the ability to select the resource adapter (connector) to execute a remote procedure call (RPC) on a given M system. On the J2EE side, the way to retrieve a connector is with a Java Naming and Directory Interface (JNDI) name. The institution, station, and division number are the pieces of information an application is likely to have to identify an M system it wants to connect to. Applications should not have to hard-code connector JNDI names; in most cases they should get the appropriate connector using a station number.

4.1. Managing the Mapping

4.1.1. PrimaryStation Attribute (VistALink Configuration File)

The primaryStation attribute for connector entries in VistALink's configuration file provide the mapping between connector JNDI names and VistA station numbers. The (required) **primaryStation** attribute of the **<connector>** element links a particular connector (identified by its JNDI name) with a particular station number.

In the following configuration file example, station number "11000" (and its descendants "11000A," "110000B," etc.) are mapped to the JNDI name "vlj/testconnector":

```
<?xml version="1.0" encoding="UTF-8"?>

<connectors encryptionScoped="false"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="connectorConfig.xsd"
xmlns="http://med.va.gov/vistalink/adapter/config"
>
   <connector
      jndiName="vlj/testconnector"
      primaryStation="11000"
      ip="test.somewhere.va.gov" port="8000" access-code="joe.123" verify-
code="ebony.432" timeout="30" always-use-default-as-min="false" enabled="true"
   />

</connectors>
```

Figure 4-1. Institution Mapping Association in VistALink Configuration File

Assuming the above connector is deployed on a given server, when an application asks for the JNDI name for the connector for station 11000A, it will be returned the string "vlj/testconnector."

4.1.2. How the Mapping Is Initialized

Institution mapping is per JVM in-memory cache, based on the settings configured by the administrator in the VistALink configuration file. It is initialized when the first connector is deployed to a given server. The mappings for each connector are added in when each connector deploys. If multiple servers have connectors deployed, each server has its own copy/version of the in-memory cache.

4.1.3. Viewing the Current Mapping

To view the current set of institution-to-station number mappings on any server, use the VistALink administration console on your WebLogic administration server (if deployed). For more information see Monitoring Institution Mapping in the section *The VistALink Administration Console*.

4.1.4. Updating Institution Mappings

Mappings are loaded automatically for a connector whenever the connector is deployed. If a previous connector has been mapped to the same station number, a "last one in wins" strategy is employed, and a warning-level logger entry is logged.

Mappings are marked for removal when a connector is undeployed. The actual removal of the mappings does not happen until garbage collection runs on the server. If garbage collection has not run and a mapping is used to retrieve the JNDI name of an undeployed connector, the attempt to use of the connector will fail at the point the application attempts to retrieve the connection factory from JNDI. If garbage collection *has* run, the attempt to use of the connector will fail when no JNDI name can be returned for the given station number.

4.1.5. How Applications Use Institution Mappings

Applications are provided with an Application Program Interface (API) to retrieve a connector's JNDI name, based on a station number. For more information, see the *VistALink 1.6 Developer Guide*.

4.2. Troubleshooting Institution Mapping Issues

4.2.1. Out-of-Synch Configuration Files on Multiple Physical Servers

The VistALink configuration file (**gov.va.med.vistalink.connectorConfig.xml)** must be accessible on the JVM classpath of each server with deployed VistALink connectors. Because separate copies of the VistALink configuration file can exist for separate physical servers, it is possible that these copies may not be identical. The `primaryStation` attributes, on which the institution mapping is based, may be different.

It is the responsibility of the system administrator to keep separate physical copies of the VistALink configuration file synchronized.

4.2.2. Connector Station and M System Mismatch

The `primaryStation` attribute is used to create the mappings that applications use to retrieve the connector. It is always possible to misconfigure a connector, associating it with the wrong station number. Station mismatch checking is employed to catch such configuration errors.

When VistALink connects to an M system, the `primaryStation` attribute of the connector is passed to M. It is then checked against the DEFAULT INSTITUTION field (#217) of the KERNEL SYSTEM PARAMETERS file (#8989.3). If it doesn't match, the connection is rejected, and a `SecurityPrimaryStationMismatchException` is returned to the application.

The system administrator should monitor the VistALink administration console and log files for indicators of rejected connections due to primary station mismatches.

4.2.3. JNDI Name Configuration Mismatch

The `jndiName` attribute in the VistALink configuration file is used to associate a station number with a particular connector. However, the JNDI name under which WLS deploys the connector comes from the `<jndi-name>` element in the **weblogic-ra.xml** file. If these two values don't match, the mapping will not reflect the correct JNDI name for the connector.

Runtime checking validates whether these values are the same. If they are not, attempts to use the connector will return a `VistaLinkResourceException` to the application. In addition, the VistALink administration console will display a warning if it sees these values do not match for a given connector.

The two JNDI settings are in the **weblogic-ra.xml** deployment descriptor and the **gov.va.med.vistalink.connectorConfig.xml** file. When these settings are out of synch, the result should be an unusable connection (outright failure), rather than an incorrectly routed connection, which is harder to diagnose.

The system administrator should monitor the VistALink administration console and log files for indicators of rejected connections due to JNDI name mismatches.

4.2.4. Mappings with Undeployed Connectors (Stopped or Deleted)

Mappings are marked for removal when a connector is undeployed. However, the actual removal of the mappings does not happen until garbage collection runs on the server.

If a connector has been undeployed and garbage collection has not yet run, the attempted use of the connector will fail at the point the application attempts to retrieve the connection factory from JNDI. If garbage collection has run, the attempted use of the connector will fail when no JNDI name can be returned for the given station number. In both cases, the attempt to use the undeployed connector will fail.

4.3. Pluggable Institution Mapping Rules

NOTE: This section is of interest primarily to non-VA users of VistALink.

The implementation of institution mapping used on the J2EE side of VistALink contains rules with several VA-specific assumptions, in particular:

- Legal station numbers in VA can be 3 digits or 5+ digits with one exception (see nursing homes below).
- For station numbers assigned to the Austin Information Technology Center (AITC) only: If a station number has an alpha suffix, e.g. "200A", the numeric portion must be "200"

- For nursing homes: If a numeric station number is 4 digits, the 4th digit must be a "9"

In order to facilitate use of VistALink outside of VA, the implementation of these VA rules has been placed into a pluggable PrimaryStationRules implementation. By default, the VA-specific implementation of the station number rules is used when working with station numbers on the J2EE side. However, to have VistALink use a different rule implementation in place of the default VA-specific one:

- A new implementation of the IPrimaryStationRules interface must be provided, and
- The package/classname of the new implementation must be passed in a -D JVM argument, "gov.va.med.vistalink.primary-station-rules-class". E.g., pass -Dgov.va.med.vistalink.primary-station-rules-class=org.test.PrimaryStationRulesTest

REF: For more information on implementing a non-VA-specific institution rules class, see the *Developer's Guide*.

5. The VistALink Administration Console

5.1. Overview

The VistALink administration console runs in two modes depending on installation: either as an extension in the WebLogic console, or as a standalone Web application. In both cases, the VistALink administration console runs on the administration server for a given WebLogic configuration. It provides the following functionality:

- Monitor Connector Status (per server)
    - Connector summary list with health indicators, status
    - Connector detail, including a live call to M system
    - Institution mapping association list
- Configuration Editor (admin server only)
    - Edit connector configurations in VistaLink connector configuration file

5.2. Monitor Role Restrictions

WebLogic's console supports four roles: Administrators, Deployers, Operators and Monitors. The 1.6 VistALink administration console restricts users in the Monitors role to read-only access. Users in the Monitors role can monitor connector status, but are restricted from using the Configuration Editor.

NOTE: To support access by users in the Monitors role, the system administrator currently needs to explicitly declare the "Listen Address" attribute in the WebLogic domain configuration, for each server in the domain. Otherwise a JMX remoting call that is needed, fails, under Monitor privileges alone.

5.3. Accessing the VistALink Administration Console

5.3.1. VistALink Administration Console as Standalone Web Application

If installed as a standalone Web application, you can access it at the following URL:

http://<admin server>:<port>/vlconsole

You'll be prompted for a user name and password. Use the same credentials as you would use to login to the WebLogic administration console. From that point on, the standalone VistALink console application will look almost identical to the console extension plug-in version (except it will not have the WebLogic console content displayed on the left and top.).

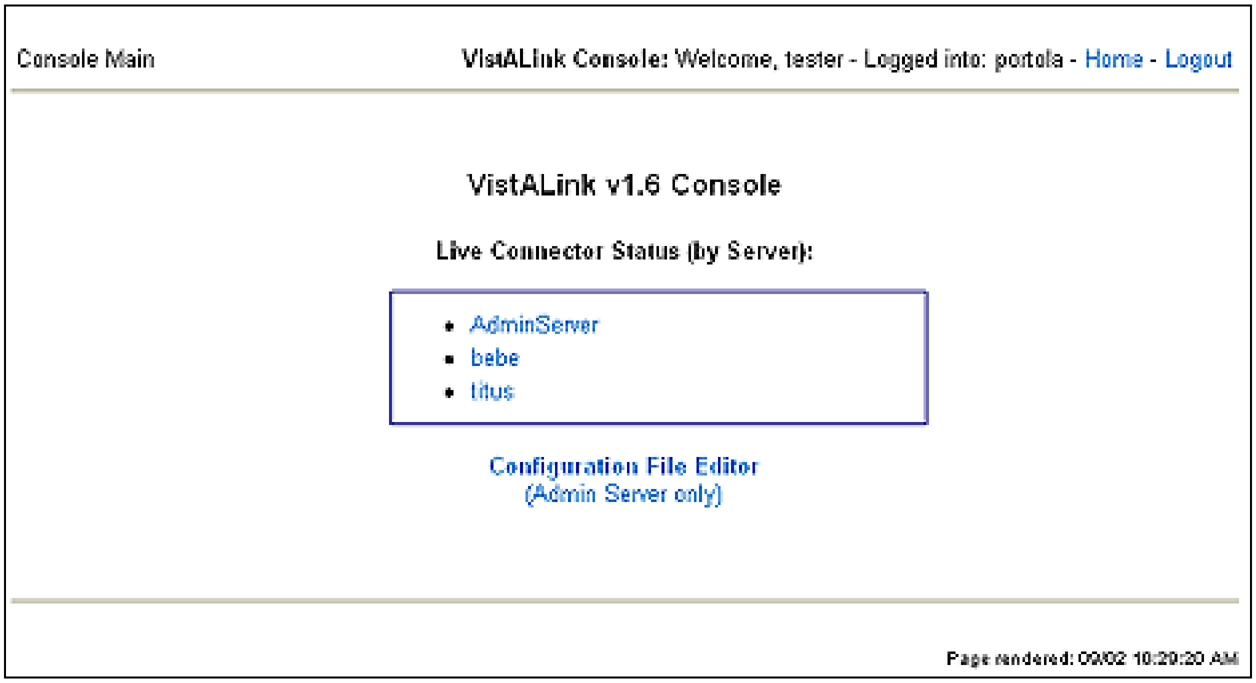

Console Main
VistALink Console: Welcome, tester - Logged into: portola - Home - Logout
VistALink v1.6 Console
Live Connector Status (by Server):
AdminServer
bebe
titus
Configuration File Editor (Admin Server only)
Page rendered: 09/02 10:29:20 AM

Figure 5-1. Standalone VistALink 1.6 Administration Console

NOTE: On WebLogic 10.3, the VistALink administration console works best when installed as a standalone application.

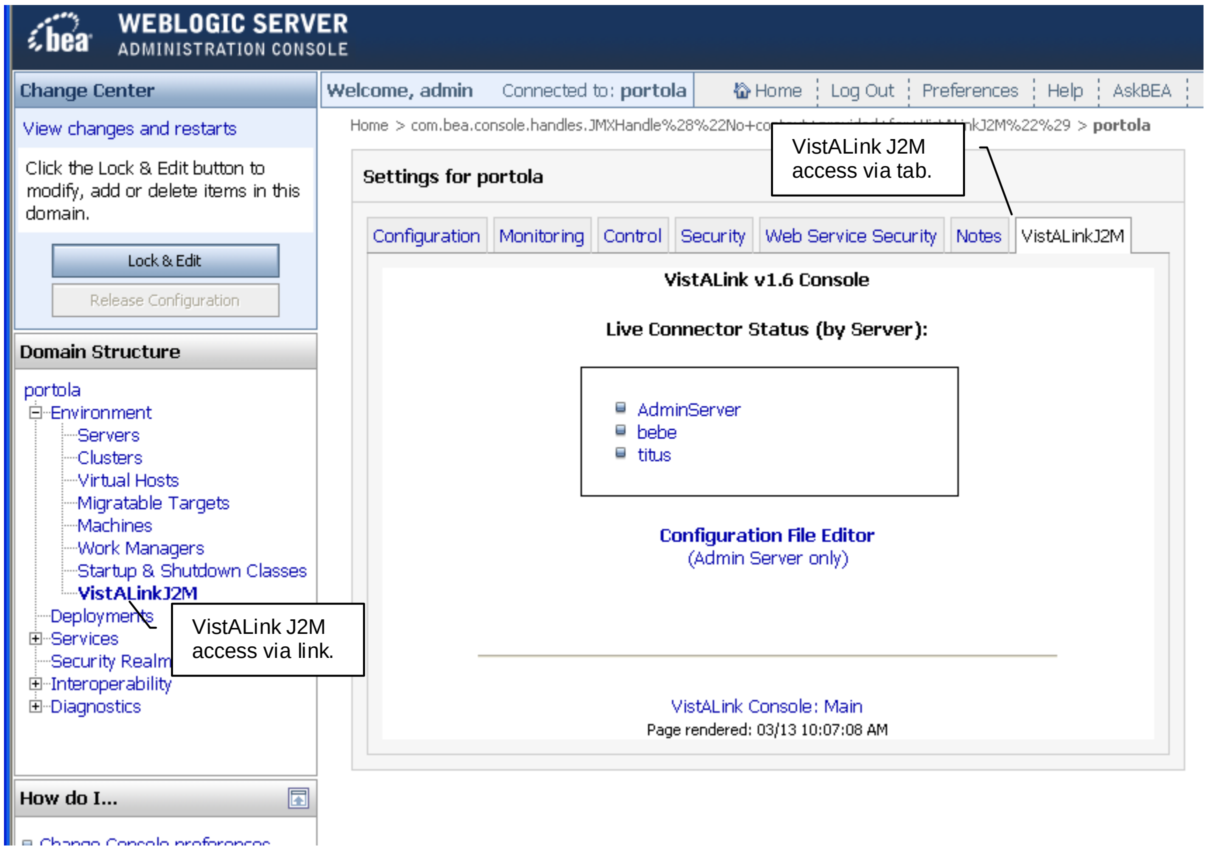
5.3.2. VistALink Administration Console as WebLogic Console Extension

If installed as an extension in the WebLogic console, access the VistALink console extension as part of the main WebLogic console. Within the WebLogic console, the VistALink console extension can be found in two locations:

- As a tab (‘VistALink J2M’) in the WebLogic console’s domain page
- As a link (‘VistALink J2M’) in the WebLogic console’s navigation tree, under ‘Environment’

NOTE: On WebLogic 10, only the tab in the WebLogic console’s domain page may be available. The link in the console’s navigation tree may not be displayed.

Figure 5-2 shows two ways to access the VistALink administration console when installed as a WebLogic console extension.



Click the Lock & Edit button to modify, add or delete items in this domain.

VistALink v1.6 Console

Live Connector Status (by Server):

Page rendered: 03/13 10:07:08 AM

Figure 5-2. VistALink Administration Console: Access via *link* in navigation tree and *tab* in domain tab set

5.4. Monitoring Connector Status

To monitor connector status, pick a server in your domain from the list of servers on the main VistALink Administration console page. You can then monitor various attributes of the VistALink adapter(s) deployed on that server.

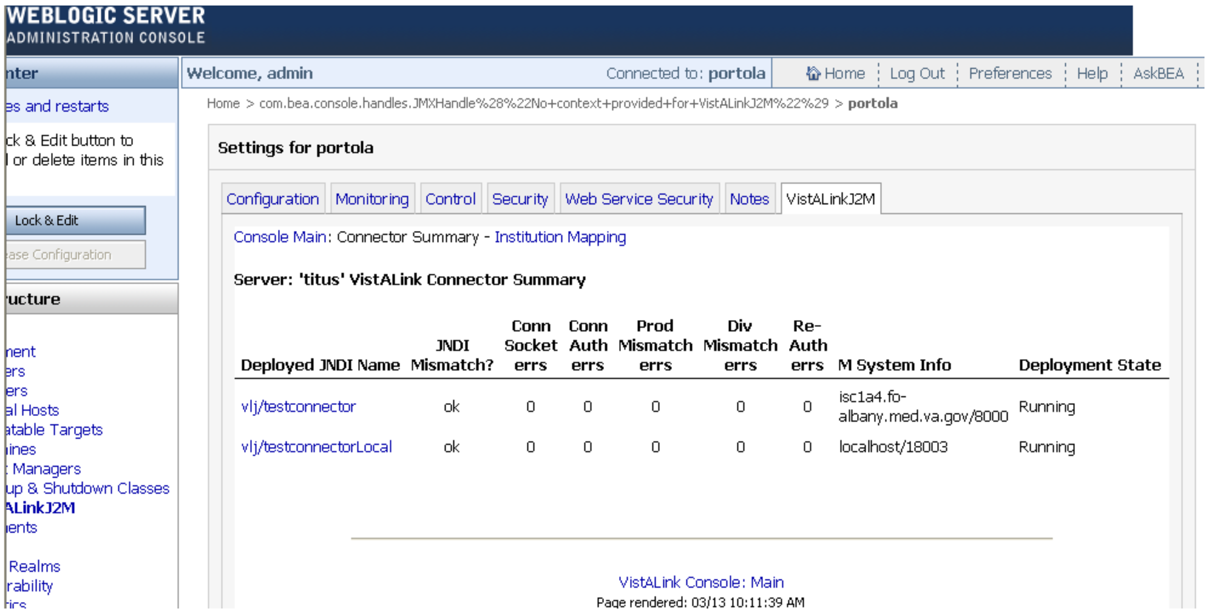
5.4.1. Adapter List/Summary

On the first page after selecting a server, Figure 5-3, you'll be shown a list of VistALink adapters on the server, including health indicators and other summary information for each:

- **Deployed JNDI Name:** The JNDI name under which the adapter has been deployed and made available to other applications
- **JNDI Mismatch:** Indicates that the JNDI name used to look up the connector settings in the VistALink configuration file (from the "connectorJndiName" custom property value in **weblogic-ra.xml**) does not match the deployed JNDI name used by the container to deploy the connector, which could indicate a misconfigured connector.
- **Health Indicators/Failure Counts:**
    - o **Conn Socket:** The number of times, since deployment or server re-start, that an attempt to create a physical connection to the M system has failed.

- **Conn Auth:** The number of times, since deployment or server restart, that the connector has attempted and failed to log on to the M system with the connector user access/verify code.
- **Prod Mismatch:** The number of times since deployment or server restart that the connector has attempted to log on to the M system and found a mismatch between the Production settings for the M system and the J2EE server's JVM. On the J2EE server, this setting is controlled via the **gov.va.med.environment.production** JVM argument.
- **Div Mismatch:** The number of RPC requests since deployment or server restart that have failed because the division specified in the RPC request is not legal for the re-authentication user. A large number of division mismatches may indicate a misconfigured JNDI-to-institution mapping for the connector in question.
- **Re-Auth:** The number of times since deployment or server restart that re-authentication has failed when attempting to execute RPCs on behalf of users. A large number of failures may indicate a misconfigured JNDI-to-institution mapping for the connector in question.

- **M System Info:** The IP address and port used to connect to the M system configured for the adapter.
- **Deployment State:** The deployment state of the adapter on the J2EE system.



WEBLOGIC SERVER
ADMINISTRATION CONSOLE
Welcome, admin
Connected to: portola
Home | Log Out | Preferences | Help | AskBEA
Lock & Edit
Configuration | Monitoring | Control | Security | Web Service Security | Notes | VistALinkJ2M

Home > com.bea.console.handles.JMXHandle%28%22No+context+provided+for+VistALinkJ2M%22%29 > **portola**

Settings for portola

Console Main: Connector Summary - Institution Mapping

Server: 'titus' VistALink Connector Summary

Deployed JNDI Name	JNDI Mismatch?	Conn Socket errs	Conn Auth errs	Prod Mismatch errs	Div Mismatch errs	Re-Auth errs	M System Info	Deployment State
vlj/testconnector	ok	0	0	0	0	0	isc1a4.fo-albany.med.va.gov/8000	Running
vlj/testconnectorLocal	ok	0	0	0	0	0	localhost/18003	Running

VistALink Console: Main

Page rendered: 03/13 10:11:39 AM

Figure 5-3. Connector Summary List

5.4.2. Adapter Detail

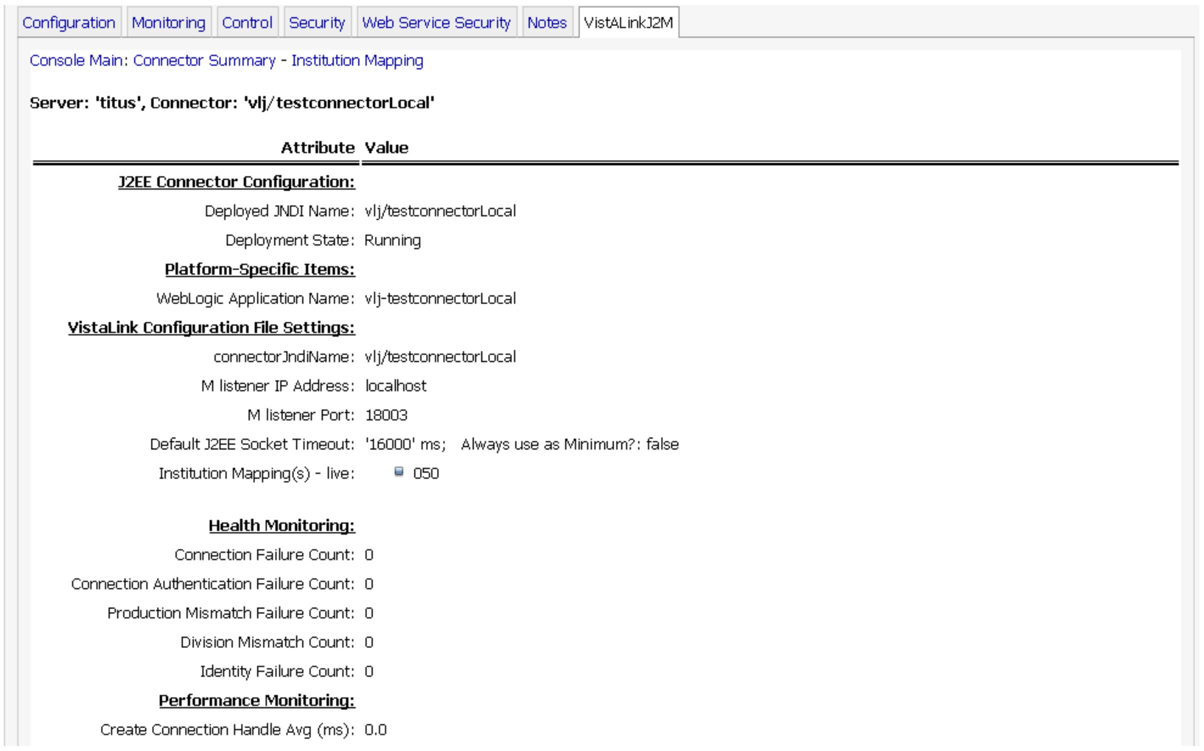
If you click on the "Deployed JNDI Name" link for any adapter on the summary page, you'll access a detail page, Figure 5-4 and Figure 5-5, with more information for the connector:

- WebLogic-specific configuration information
- VistALink-specific configuration information

- including institution mappings associated with the connector

- Health monitoring indicators
- Performance monitoring indicators
- Live VistALink M/VistA Server Query, including (from M):
  - M VistALink version
  - "Connector Proxy" user name from the New Person file
  - UCI/volume/box-volume pair of the M account
  - Domain, Institution and Test/Production setting of the M account
  - Introductory Text (for visual confirmation of correct system)

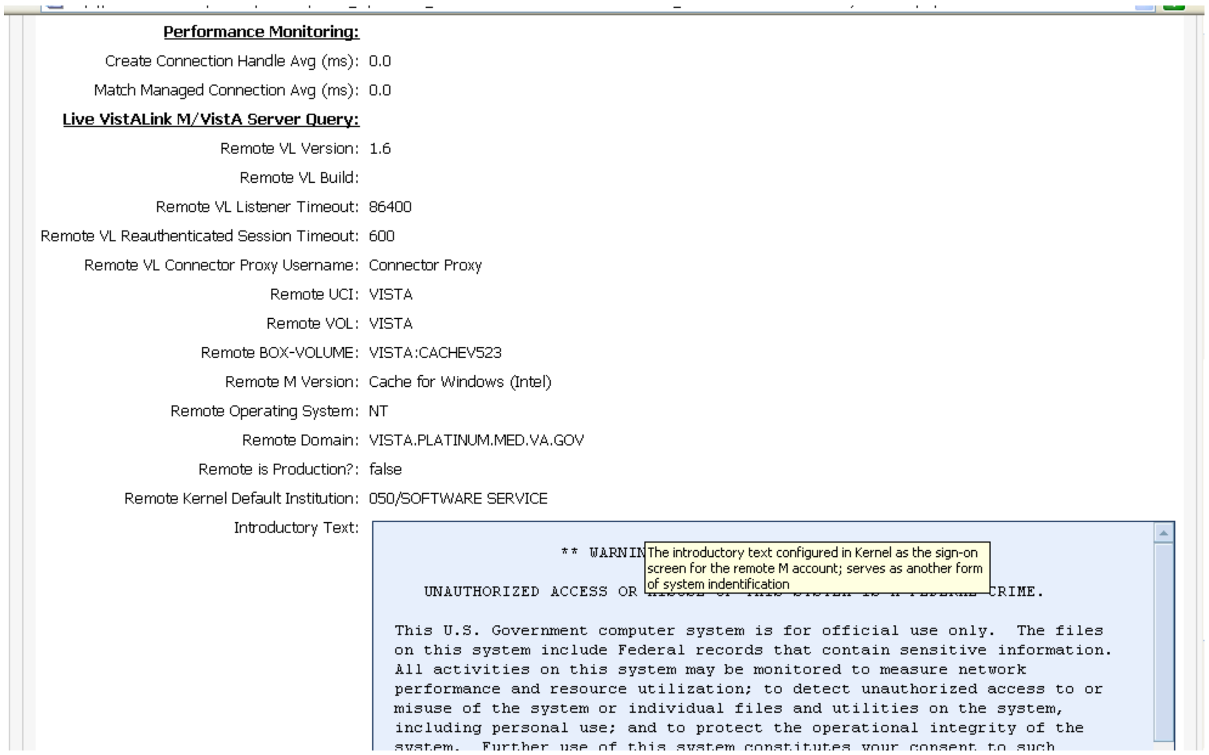
NOTE: When the detail page is loaded for any connector, a "live" call is made to the M system. The results provide an easy way to get a picture of the M system connected to, and may also help verify whether a connector is reaching the *intended* M system.



Server: 'titus', Connector: 'vlj/testconnectorLocal'

Attribute	Value
J2EE Connector Configuration:	
Deployed JNDI Name:	vlj/testconnectorLocal
Deployment State:	Running
Platform-Specific Items:	
WebLogic Application Name:	vlj-testconnectorLocal
VistaLink Configuration File Settings:	
connectorJndiName:	vlj/testconnectorLocal
M listener IP Address:	localhost
M listener Port:	18003
Default J2EE Socket Timeout:	'16000' ms; Always use as Minimum?: false
Institution Mapping(s) - live:	▪ 050
Health Monitoring:	
Connection Failure Count:	0
Connection Authentication Failure Count:	0
Production Mismatch Failure Count:	0
Division Mismatch Count:	0
Identity Failure Count:	0
Performance Monitoring:	
Create Connection Handle Avg (ms):	0.0

Figure 5-4. Connector Detail, 1 of 2



Performance Monitoring:

Create Connection Handle Avg (ms):	0.0
Match Managed Connection Avg (ms):	0.0

Live VistALink M/VistA Server Query:

Remote VL Version:	1.6
Remote VL Build:	
Remote VL Listener Timeout:	86400
Remote VL Reauthenticated Session Timeout:	600
Remote VL Connector Proxy Username:	Connector Proxy
Remote UCI:	VISTA
Remote VOL:	VISTA
Remote BOX-VOLUME:	VISTA:CACHEV523
Remote M Version:	Cache for Windows (Intel)
Remote Operating System:	NT
Remote Domain:	VISTA.PLATINUM.MED.VA.GOV
Remote is Production?:	false
Remote Kernel Default Institution:	050/SOFTWARE SERVICE

Introductory Text:

```
                    ** WARNIN

   UNAUTHORIZED ACCESS OR                                          CRIME.

 This U.S. Government computer system is for official use only.  The files
 on this system include Federal records that contain sensitive information.
 All activities on this system may be monitored to measure network
 performance and resource utilization; to detect unauthorized access to or
 misuse of the system or individual files and utilities on the system,
 including personal use; and to protect the operational integrity of the
 system.  Further use of this system constitutes your consent to such
```

The introductory text configured in Kernel as the sign-on screen for the remote M account; serves as another form of system indentification

Figure 5-5. Connector Detail, 2 of 2

5.4.3. Monitoring Institution Mapping

Click on “Institution Mapping” to display the current set of institution mapping associations on the selected server. For example:

```
Station #  JNDI Name
11000      vlj/testconnector
11001      vlj/testconnector1
11002      vlj/testconnector2
```

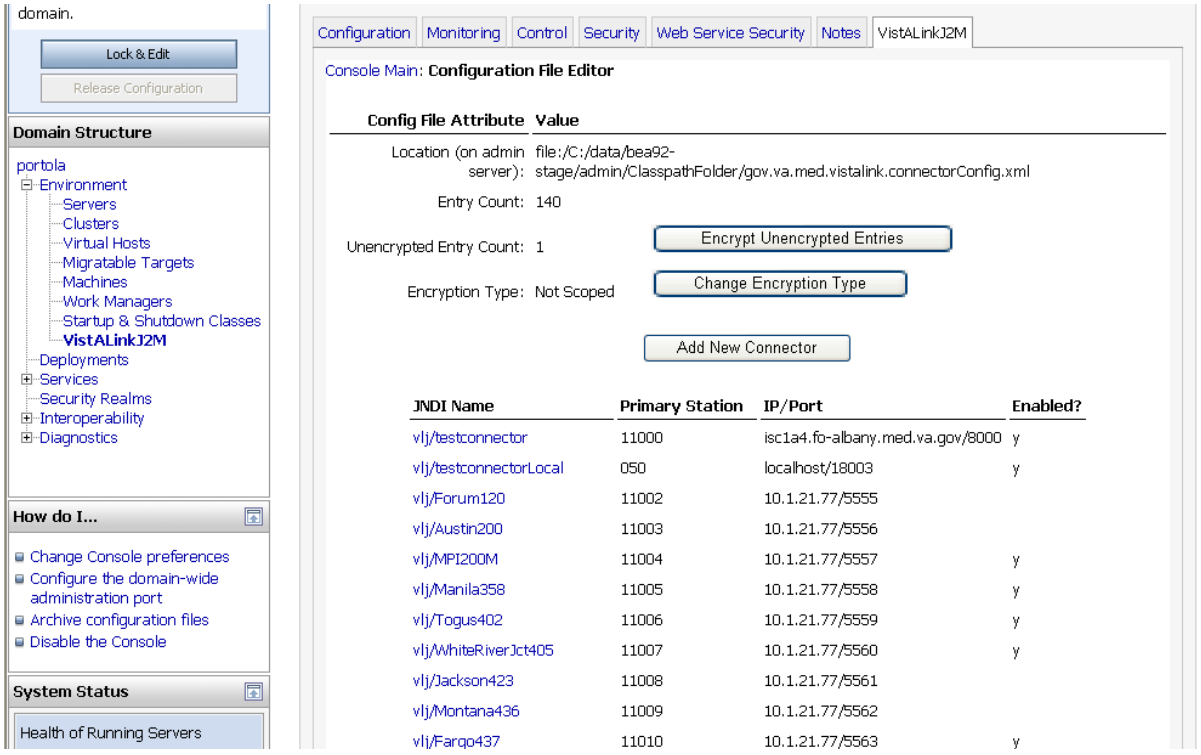
The current mappings for the selected server are listed in a table. You can use this information to monitor and verify the current, live institution mappings on your server(s).

NOTE: Applications typically retrieve an adapter by using the station #. The actual adapter returned to the application is determined by what JNDI name is mapped to the requested station #. The list of mappings is generated (and modified) when each adapter’s configuration is loaded during adapter deployment, update, or server startup.

5.5. Configuration Editor

The VistALink administration console includes the Configuration Editor, for editing the VistALink configuration file storing information about each VistALink adapter/connector, Figure 5-6.

NOTE: Users that are in the Monitor role (and not a Deployer, Operator or Administrator) are disallowed from using the configuration editor.



Console Main: **Configuration File Editor**

Config File Attribute	Value
Location (on admin server):	file:/C:/data/bea92-stage/admin/ClasspathFolder/gov.va.med.vistalink.connectorConfig.xml
Entry Count:	140
Unencrypted Entry Count:	1
Encryption Type:	Not Scoped

JNDI Name	Primary Station	IP/Port	Enabled?
vlj/testconnector	11000	isc1a4.fo-albany.med.va.gov/8000	y
vlj/testconnectorLocal	050	localhost/18003	y
vlj/Forum120	11002	10.1.21.77/5555	
vlj/Austin200	11003	10.1.21.77/5556	
vlj/MPI200M	11004	10.1.21.77/5557	y
vlj/Manila358	11005	10.1.21.77/5558	y
vlj/Togus402	11006	10.1.21.77/5559	y
vlj/WhiteRiverJct405	11007	10.1.21.77/5560	y
vlj/Jackson423	11008	10.1.21.77/5561	
vlj/Montana436	11009	10.1.21.77/5562	
vlj/Fargo437	11010	10.1.21.77/5563	y

Figure 5-6. Configuration Editor Main Interface

5.5.1. About the Configuration File

The configuration file contains additional properties needed to define a VistALink adapter beyond those in the deployment descriptors (ra.xml and weblogic-ra.xml).

The Configuration Editor loads the configuration file from the admin server classpath. If a file named "gov.va.med.vistalink.connectorConfig.xml" is found in a folder on the admin server classpath, the Configuration Editor loads the contents of this file automatically. When changes are made and saved, they are saved to this file as well, on the admin server only.

NOTE: The main page of the Configuration Editor displays the physical file location of the VistaLink connector configuration file that it is using, on the admin server.

Configuration files must conform to the schema file **connectorConfig.xsd**, provided with the VistALink 1.6 distribution. For more information about the VistALink configuration file, see the section, "VistALink Connector Configuration File."

5.5.2. How to Propagate Configuration Changes to Other Servers

A copy of the configuration file must be available on the classpath of all servers in a domain that have deployed VistALink adapters. Changes made by the Configuration Editor, however, are made to the copy of the file on the physical file system of the admin server only.

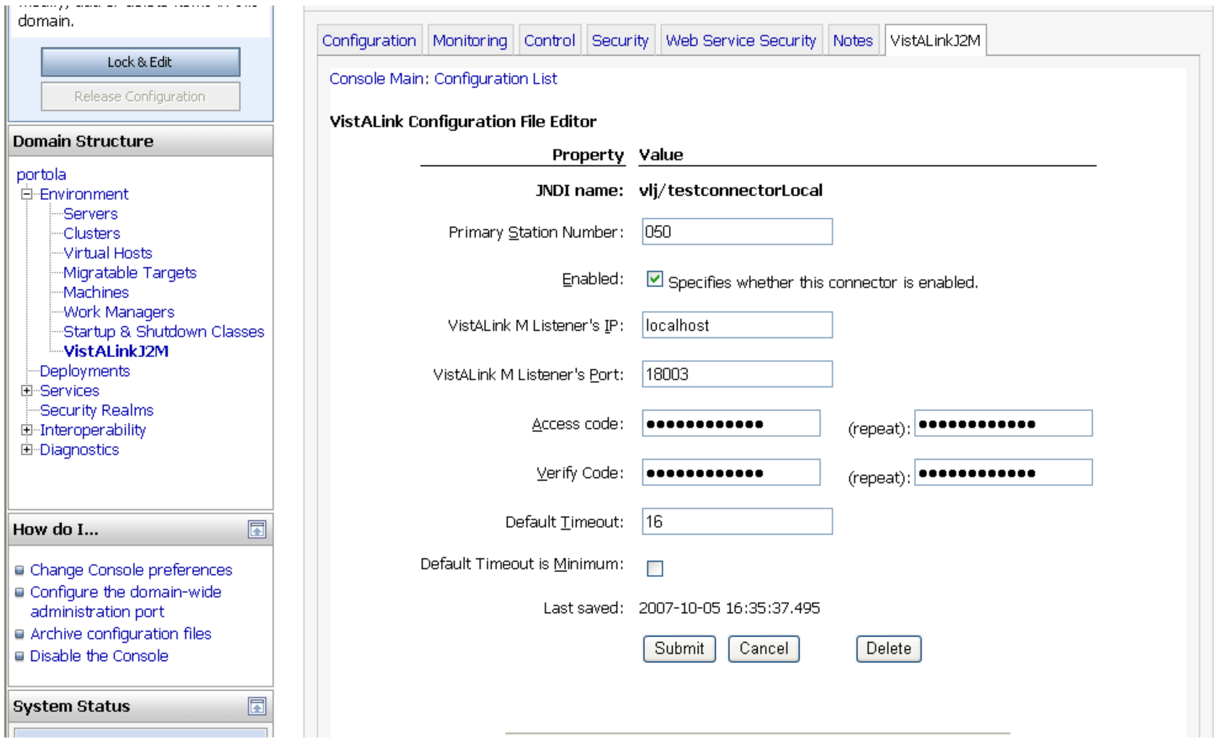
Multiple Server, Multiple Machine Domain: In order to update the per-server VistALink configuration files for other physical servers in your domain (i.e., managed servers), you must manually copy/propagate the edited file from the admin server to a location on the classpath of the particular server.

Multiple Server, Single Machine Domain: Because all of the servers in this type of domain reside on the same physical machine, it is possible to configure all of their classpaths to contain the same folder, holding a single copy of the VistALink configuration file. In such cases, no further propagation of the configuration file is required.

Single Server Domain: In a single server WebLogic domain, one server is both the admin server and the server on which connectors and applications are deployed. No further propagation of the configuration file beyond the admin server is required.

5.5.3. Viewing and Editing Connector Properties

Users can view and edit connector properties by selecting one of the connectors (hyperlinked) from the connector list. The Configuration Editor will then display the connector's properties, Figure 5-7.



VistALink Configuration File Editor

Property	Value		
JNDI name:	vlj/testconnectorLocal		
Primary Station Number:	050		
Enabled:	✓ Specifies whether this connector is enabled.		
VistALink M Listener's IP:	localhost		
VistALink M Listener's Port:	18003		
Access code:	●●●●●●●●●●●●●	(repeat):	●●●●●●●●●●●●●
Verify Code:	●●●●●●●●●●●●●	(repeat):	●●●●●●●●●●●●●
Default Timeout:	16		
Default Timeout is Minimum:	☐		
Last saved:	2007-10-05 16:35:37.495		

Figure 5-7. Editing a Connector Entry

After editing the file, you can save the changes by clicking the **Submit** button.

For a description of the various properties in a connector configuration, see the earlier section in the *Deploying VistALink on J2EE* chapter, "VistALink Connector Configuration File."

5.5.3.1. JNDI Naming Recommendations

When naming adapters and editing connector entries, we recommend that you begin all connector JNDI names with a common JNDI subcontext name, i.e., "vlj/", followed by a meaningful name to specifically identify the adapter, i.e., "vlj/Salem658".

The Configuration Editor permits the following punctuation characters as part of the JNDI name (the first letter of the JNDI name must be alphanumeric, however): - _ / \ ( ) [ ]

5.5.4. Encryption of Access/Verify Codes

Encryption of the access and verify code adds an additional level of security to protect connector proxy user credentials. The properties are encrypted when you:

- Save change when you edit an existing connector, or add a new connector
- Select "Encrypt Unencrypted Entries" on the main Configuration Editor page (only shown if there are unencrypted entries in the file)
- When a connector configuration is loaded by a deployed adapter on a given system

The main page of the Configuration Editor displays the number of unencrypted connector entries, if any.

5.5.4.1. Modifying Access/Verify Codes Outside of Configuration Editor

If necessary, the access and verify code can be modified outside the Configuration Editor, and stored in clear text. When doing this, set the optional **encrypted** attribute to "false". Then the next time any of the encryption-triggering event above happen for the connector entry, the access and verify codes will be encrypted and the **encrypted** attribute set to "true".

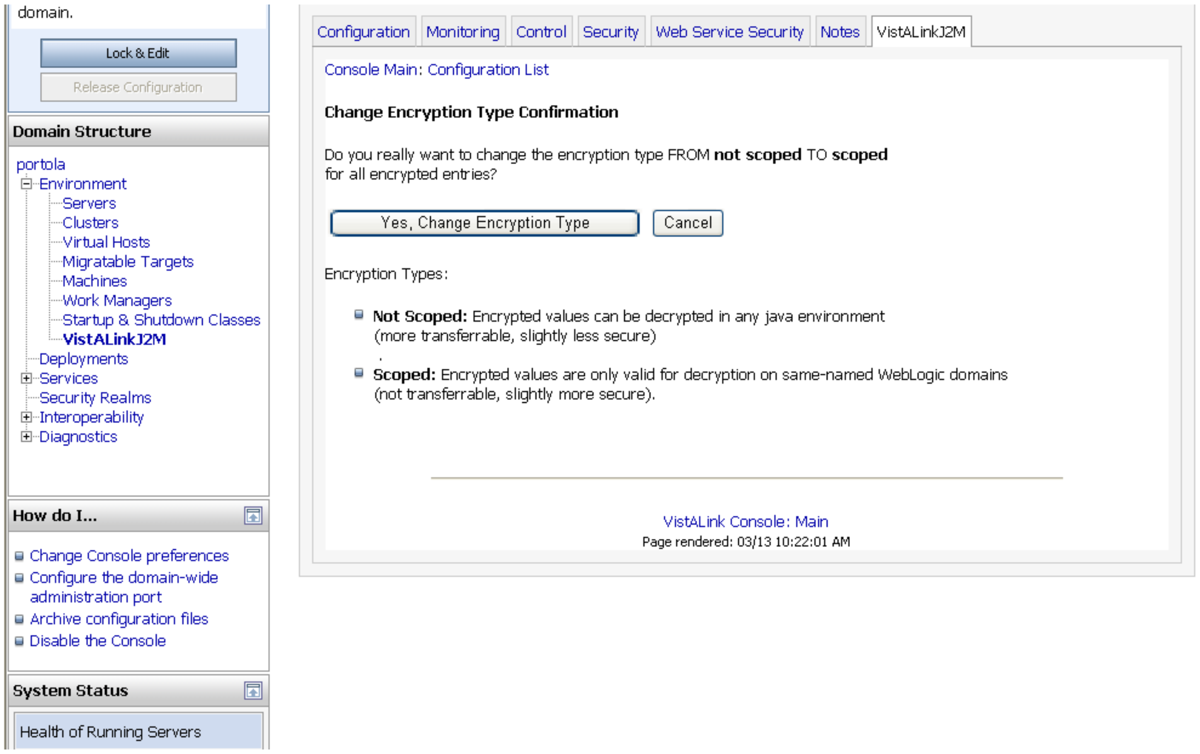
5.5.4.2. Encrypt Unencrypted Entries Feature

If any unencrypted entries are present in the configuration file, a 'Encrypt Unencrypted Entries' button is displayed. Choosing this will allow you to encrypt all unencrypted entries present in the file.

5.5.4.3. Changing the Encryption Type

The encryption type reflects whether entries have been encrypted with domain-scoped encryption or not. If domain scoped, the domain name is included in the encryption, making it difficult to use encrypted values on other WebLogic domains. Configuration files from before 1.6 do not have domain-scoped encryption.

To change the encryption type, select Change Encryption Type on the main Configuration Editor page, Figure 5-8.



Change Encryption Type Confirmation

Do you really want to change the encryption type FROM **not scoped** TO **scoped** for all encrypted entries?

Encryption Types:

- **Not Scoped:** Encrypted values can be decrypted in any java environment (more transferrable, slightly less secure)
.
- **Scoped:** Encrypted values are only valid for decryption on same-named WebLogic domains (not transferrable, slightly more secure).

Page rendered: 03/13 10:22:01 AM

Figure 5-8. Changing the encryption type -- confirmation page

6. Monitoring VistALink via JMX and MBeans

6.1. Overview

Java Management eXtensions (JMX) is the Java-standard mechanism for making applications manageable, including providing monitoring capabilities. Applications (and the application server itself) supply a variety of MBeans. Through JMX, attributes on the MBeans can be monitored, and operations invoked, both locally inside a JVM and remotely from outside the JVM.

A variety of built-in and third-party utilities and applications support the use of JMX to access MBean attributes and operations, including:

- third-party JMX consoles
- third-party monitoring applications
- scripting tools such as WebLogic Scripting Tool (WLST) and wlshell
- the WebLogic console itself, which is built around JMX and MBeans

6.2. VistALink MBeans

VistALink provides two MBean types that can be monitored:

6.2.1. VistaLinkConnector MBean

Deployed: One instance per JVM, per deployed VistALink adapter
MBean Type: VistaLinkConnector

VistaLinkConnector Attribute	Type	Description
CfgIpAddress	java.lang.String	IP address from VistALink configuration file used for this connector.
CfgPort	int	port from VistALink configuration file used for this connector
CfgTimeout	long	timeout from VistALink configuration file used for this connector
CfgTimeoutAlwaysUseDefaultAsMin	boolean	whether a lower timeout can be set by applications than the configured timeout, from VistALink configuration file used for this connector
ContainerMBeanName	javax.management.ObjectName	System-specific ObjectName of connector's connector MBean linked to this VistALink MBean
DeploymentState	java.lang.String	the current deployment state of the connector (platform-specific string) Throws: java.rmi.RemoteException -

VistaLinkConnector Attribute	Type	Description
		thrown if unable returns this information (obtained from another MBean)
DistinguishedIdentifier	long	unique, internally managed distinguished identifier for the VistALink connector
EisType	java.lang.String	the EISType of the connector (for VistALink connectors, should always be 'VistA') Throws: java.rmi.RemoteException - thrown if unable returns this information (obtained from another MBean)
HlthConnectionAuthFailureCount	long	number of connector proxy authentication failures for this connector, since last system startup or deployment
HlthConnectionFailureCount	long	number of TCP-level connection failures for this connector, since last system startup or deployment
HlthDivisionMismatchCount	long	number of mismatches between connector primaryStation setting and target M system's primary station, since last system startup or deployment
HlthIdentityFailureCount	long	number of re-authentication identification failures (failure to match requested DUZ, Application Proxy or VPID on M system), since last system startup or deployment
HlthProductionMismatchCount	long	number of mismatches between J2EE server's and M server's production setting, since last system startup or deployment
JndiNameActual	java.lang.String	the JNDI name used to register the connector in the JNDI tree of the server
JndiNameCustomProp	java.lang.String	the value of the connectorJndiName custom ra.xml property
PerfCreateConnectionHandleAvgMillis	double	The amount of time it takes for a VistaLink managed connection to create a new connection handle for the underlying physical connection represented by the ManagedConnection instance; this connection handle is used by the application code to refer to the underlying physical connection
PerfMatchManagedConnectionAvgMillis	double	The amount of time it takes for the VistaLink connection factory to either a) match a connection request with an

VistaLinkConnector Attribute	Type	Description
		existing connection in pool of managed connections, or b) determine that no such match exists
QueryMappedInstitutions	java.lang.String[]	a formatted string array listing the institutions currently mapped to this connector
QueryMSystemMap	java.util.Map	returns a Map containing a set of string keys and values obtained from calling the M system.
QueryMSystemReport	java.lang.String[]	a formatted string array report representing a set of properties and values obtained from calling the M system
VendorSpecificProperties	java.util.Map	a Map containing vendor-specific properties and values Throws: java.rmi.RemoteException - thrown if unable returns this information (obtained from other MBeans)

Table 6-1. VistaLinkConnector MBean Attributes

6.2.2. VistaLinkInstitutionMapping MBean

- Deployed: One instance per JVM where VistALink Connectors are Deployed
- Type: VistaLinkInstitutionMapping

Attribute Name	Type	Attribute
InstitutionMappings	java.util.Map	A Map containing string keys (station numbers) and string values (JNDI names) representing the current institution mapping associations on this JVM.

Table 6-2. VistaLinkInstitutionMapping MBean Attributes

6.3. VistALink MBean Security

By default, in WebLogic 9/10 domains, customer-supplied MBeans such as those provided by VistALink are secured as follows:

- attributes: anyone who can authenticate to the domain can access
- operations: must possess administrator role to invoke

All VistALink MBean information access is implemented as attributes.

7. M Server Management

7.1. Overview

Because VistALink connectors are deployed on J2EE servers, much of the management workload for VistALink focuses on the J2EE server. However, the VistALink listener (the portion of VistALink that resides on the M server) must also be managed.

In some cases, the same system manager may be responsible for managing both the J2EE and M portions of VistALink. In many cases, however, the two sides will be managed by different system managers, in separate computing facilities.

7.2. Finding User Processes with the Connection Manager

The Connection Manager (new with VistALink 1.6) tracks all active VistALink connections on the current M system, displays the individual processes, and gives you the option to either kill a selected VistALink connection, or all VistALink connections. The Connection Manager could be useful during software installations, for example, where it might be desirable to get all VistALink processes off of the system.

In multi-node M environments, the M node you run the Connection Manager on should be on the same node as the VistALink connections are running on, as started by the VistALink listener(s).

Main screen capture:

```
VL/J2M Connection Manager     Feb 21, 2008@11:33:14         Page:    1 of    1
    Job Selection Criteria (matches: 4)
     Box-Volume Pair: ROU:CACHE2
   Current Namespace: VL-HEAD
             Routine: XOBVSKT
           Job State: READ :: Job is reading from a device.
 Entry PID        Device:Client IP                   Comment
    1  1756       |TCP|8016|1756:10.6.21.65          User=*KILMADE,JOE
    2  1757       |TCP|8016|1757:10.6.21.65          User=*KILMADE,JOE
    3  11769      |TCP|8016|11769:10.6.17.157        User=*KILMADE,JOE
    4  11770      |TCP|8016|11770:10.6.17.157        User=*KILMADE,JOE




          * Connector Proxy User
TA  Terminate All                       RE  Refresh
TP  Terminate PID                       SS  System Status
Select Action:Quit//
```

Figure 7-1. Using the Connection Manager

Action	Description
TA (Terminate All)	Kills all VistALink connections
TP (Terminate PID)	Kills a selected VistALink connection
RE (Refresh)	Refreshes/updates the list of connections
SS (System Status)	Displays a list of all processes running on the system (not just VistALink ones)

Table 7-2. Connection Manager actions and definitions

7.3. Finding VistALink Processes without the Connection Manager

7.3.1. VMS Systems

On VMS systems, the listener for any active VistALink connection is typically configured to be launched via VMS TCP Services. VistALink sets the VMS process name to "`VLink_XXXXXXXX`", where "XXXXXXXX" is the value of the M $J converted to hex.

This makes it easier to find which VMS processes are associated with VistALink, and with which M job as well. You can use the DCL command **SHOW SYSTEM /PROC=VL** to display all VistALink processes. For example:

```
core$ show sys /proc=VL* <RET>
OpenVMS V7.3-2  on node ISC1A4   1-MAR-2005 10:19:47.64  Uptime  146 10:17:35
  Pid    Process Name    State  Pri      I/O       CPU       Page flts  Pages
20F9B10B VLINK_BG1877    HIB     11      179   0 00:00:00.01       202    184  N
20F9F90C VLink_20F9F90C  LEF     11      211   0 00:00:00.11      4186    852  S
20FA190D VLINK_BG1891    HIB     11      172   0 00:00:00.04       202    184  N
20F9890E VLink_20F9890E  LEF     10      211   0 00:00:00.10      4310    865  S
20F9C90F VLINK_BG1910    HIB     11      172   0 00:00:00.01       202    184  N
20F9C910 VLink_20F9C910  LEF     10      211   0 00:00:00.09      4231    852  S
core$
```

Figure 7-3. Use the DCL command SHOW SYSTEM /PROC=VL to display all VistALink processes

7.3.2. Non-VMS Systems

To list VistALink processes on non-VMS systems (e.g. Caché on Windows), you can look for processes that are frequently in the XOBVSKT routine; these processes are VistALink listeners. (However, when RPCs are invoked, the routine listed will be different.)

7.4. Listener Management

7.4.1. Listener Management for Caché/VMS Systems

See the section "Listener Management for Caché/VMS" in Chapter 2 ("VistA/M Server Installation Procedures") of the *VistALink 1.6 Installation Guide*.

7.4.2. Listener Management for Caché/NT Systems

See Appendix A, "Listener Management for Caché/NT Systems."

7.4.3. Listener Management for DSM/VMS Systems

See Appendix B, "Listener Management for DSM/VMS Systems."

7.5. M Listener Site Parameters File

The FOUNDATIONS SITE PARAMETERS file (#18.01) is used to control listener settings for all implementations (Caché and DSM, on VMS and Windows). It contains one entry; the ".01" field is a pointer to the DOMAIN file (#4.2). When VistALink is installed, the install process creates this entry and assigns the proper Domain Name using the DOMAIN file.

The site parameters in this top-level entry pertain to Foundations and VistALink. Currently, all the parameters in this file are related to VistALink and listener configuration. As more Foundations tools are introduced, non-VistALink-related parameters will be added.

NOTE: As part of the VistALink installation for Caché systems, a listener configuration named 'DEFAULT' was created for port 8000.

To edit the site parameters, use the Site Parameters action in the Foundations menu.

7.5.1. Site Parameters for All System Types

To edit VistALink related site parameters, use the Site Parameters action:

```
HEARTBEAT RATE: 180// <Enter>
LATENCY DELTA: 180// <Enter>
J2EE CONNECTION TIMEOUT: 604800//
J2EE REAUTHENTICATION TIMEOUT: 3600//
```

Figure 7-4. Edit VistALink related site parameters

7.5.2. Site Parameters for Windows Systems

When the listener runs in M (rather than via VMS TCP Services), you should make an entry in the VistALink LISTENER CONFIGURATION file for each set of listeners that you plan to run on your system. After adding a listener configuration, you need to associate that configuration with any BOX-VOLUME where the new configuration is appropriate.

Note that the following portion of the dialog is not presented to a DSM system user:

```
Select BOX-VOLUME PAIR: ROU:CACHE//
  BOX-VOLUME PAIR: ROU:CACHE// <Enter>
  DEFAULT CONFIGURATION: DEFAULT// <Enter>
Select BOX-VOLUME PAIR:
```

Figure 7-5. Edit site parameters for Windows systems

It is irrelevant on Caché /VMS as well, if the listener is started via VMS TCP Services.

As part of this action, you are asked to select a Box-Volume Pair entry. Then, within each Box-Volume Pair entry (representing the volume set and system on which the listener should run), you can set the default listener configuration to be automatically started as part of the execution of the XOBV LISTENER STARTUP option. Also, the Start Box action uses this default listener configuration.

The table below defines the Foundations Site Parameter file fields.

Field	Description
Heartbeat Rate	This field indicates the rate (in seconds) of the VistALink heartbeat message originating from a client. If there is no activity on the connection for the specified amount of time, the client will send a system heartbeat message. The client, as part of the initial connection protocol, retrieves this value. As a result, the client and the M server are always synchronized regarding the heartbeat rate.
Latency Delta	This field indicates the number of seconds to add to the HEARTBEAT RATE when calculating the initial timeout value for the VistALink listener. The client and the M server are synchronized regarding the HEARTBEAT RATE. This latency parameter allows the site to fine-tune the timeout value. The site can to take into account any network slowness or other factors that may delay the arrival of the system heartbeat message from the client.
J2EE Connection Timeout	This field indicates the number of seconds that a VistALink connection from J2EE to M should be allowed to remain connected when inactive, before M drops the connection. It is recommended that the J2EE CONNECTION TIMEOUT parameter be set relatively high, e.g., 1 day (86400 seconds). A high setting is recommended because all the major application server implementations have more robust mechanisms for controlling connection pools. The site should use the tools/mechanisms supplied by the application server implementation to control how the connection pool size grows and

Field	Description
	shrinks.
J2EE Re-authentication Timeout	This new parameter indicates the number of seconds between 180 and 3600 (inclusive) before a re-authenticated connection is considered expired and must go through the re-authentication process again. (default: 600 secs). Rules: • If a connection in the application server's pool is not reused by a particular re-authenticated user before this timeout limit is reached, the re-authentication is considered expired. • If the timeout is reached, the re-authentication process is performed again even if the same user uses the connection. • If the same connection is used again by the re-authenticated user before the timeout is reached, the session is considered re-authenticated for these number of seconds moving forward. (i.e. the timeout clock is reset) • If a different user gains access to the connection, the connection is immediately considered not re-authenticated, and the re-authentication process is performed. Example of Re-authentication Expiration: 1. User gains access and uses a connection from the pool at 4:00pm 2. User signs off and goes home, along with most of the site staff 3. After 4:00 pm, user file maintenance is performed and the user's profile is changed. For example, the user's FILE MANAGER ACCESS CODE [DUZ(0)] is changed. 4. User signs back on the next morning at 8 am 5. Since there is very little activity from 4:00-8 am, the connection has not been re-used by another user and is still associated with 4:00 user 6. Since timeout has passed, re-authentication has expired 7. Re-authentication process occurs for the user 8. Re-authentication process resets DUZ(0) appropriately to the new value.
Box-Volume Pair	(For M-based listeners only) This field indicates the BOX-VOLUME pair for the entry. The XOBV LISTENER STARTUP option uses this field to find the configuration that should be used to startup VistALink listeners for the BOX-VOLUME pair. **Note:** This information is presented to both VMS and NT Cache´ users, but not to DSM users. However, it is ignored if listeners are started via the VMS TCP Service.
Default Configuration	(For M-based listeners only) This field indicates the default startup listener configuration for the BOX-VOLUME PAIR entry. The XOBV LISTENER STARTUP option uses this field to retrieve the correct listener configuration from the VISTALINK LISTENER CONFIGURATIONS file (#18.03). The information in the configuration is then used to startup the indicated

Field	Description
	VistALink listeners on the desired ports. **Note:** This information is not presented to a DSM system user.

Table 7-6. Foundations Site Parameter file field definitions

8. Security

8.1. J2EE System Manager Security Tasks for VistALink

The primary J2EE system manager tasks for ensuring VistALink security are:

- Safeguarding the access/verify codes that M system managers provide to configure connectors with.

  These access/verify codes are for "connector proxy" users, and must be kept confidential. The level of system access they provide to the M system is significant.

- Ensuring that connectors are properly configured to reach the appropriate M systems.

  This is done primarily through the VistALink configuration file (**gov.va.med.vistalink.connectorConfig.xml**) and by making sure that connectors are mapped to the appropriate VA station number, IP, and Port.

8.2. M System Manager Security Tasks for VistALink

The primary M system manager tasks for ensuring security for VistALink are:

- Creating connector proxy users for J2EE systems connecting to your M system
- Securely communicating connector proxy credentials (access and verify codes) to authorized J2EE system managers, who use them to configure their J2EE systems to access your M system.

8.3. VistALink's J2SE Security Model

For J2SE clients operating in client/server mode with VistA, the VistALink security model is as follows:

1. A persistent connection is created between the Java client and the M server. Actions cannot be performed until the DUZ array has been created on the M side, acting as proof of end-user identity.
2. To create the DUZ array, the Java client prompts the user for access/verify codes, division (as needed), and new verify code (as needed), and submits this information to Kernel security.
3. If the credentials are valid, the DUZ array is created.

From that point on, the client can issue RPC requests to the M server. Authorizations are checked through the normal RPC authorization call, based on the contents of the DUZ array. If authorized, the RPC is executed and results are returned. Because this is a persistent connection, the DUZ array "persists" between calls. Because the connection is not used by or available to other clients, it does not need to be reproved each time.

8.4. VistALink's J2EE Security Model

The security modes for J2EE and J2SE are different. In J2EE n-tier mode, the connections are pooled on the J2EE server and are reused between different users. There are two levels of security.

The first security level is the authentication for the connection itself. It is implemented similarly to the client/server-mode mechanism described in the previous section:

1. The app server/connector is configured by the app server administrator with the access/verify code of an M user. The M user must be marked as a "connection proxy" user. (In J2EE mode, only users whose Kernel account is marked as "connection proxy" can have connections established on their behalf.)
2. If the credentials are valid, a J2EE connection is established.
3. The J2EE server places the connection in a connection pool, for use by J2EE applications running on the J2EE server.

In the J2CA specification, the second level of security is a process called *re-authentication*. Here, the end-user identity information is passed from the calling application to VistALink via a J2CA connection spec. The re-authentication steps are as follows:

1. When a J2EE application obtains a connection from the J2EE server's pool of connections, it passes identity information about the end-user to VistALink via a J2CA "connection spec."
2. VistALink (on the J2EE side) passes the identity (DUZ, VPID, or application proxy name) down to M.
3. Via a "chain of trust," VistALink (on the M side) trusts that the calling J2EE application has authenticated the identity of the end-user (via FatKAAT, KAAJEE, etc.).
4. VistALink (on the M side) performs a lightweight security context switch to the identity of the specified end-user. RPCs are then executed under the DUZ of the end-user, and normal RPC security is applied.
5. RPCs continue to execute under the identity of the specified end-user until the connection is "closed" (returned to the pool) by the J2EE application.

8.4.1. Connector Proxy User (J2EE)

The M server’s primary gatekeeper for controlling J2EE/VistALink access to M is the *connector proxy user*. This is a special user class in the Kernel NEW PERSON file (#200). Only J2EE connectors configured with credentials for "connection proxy" Kernel users can connect to an M VistALink listener. Once a connection is established through a connection proxy user, J2EE applications on that J2EE server can execute a variety of RPCs on the M server under a variety of end-user identities.

8.4.2. J2EE Re-authentication Mechanisms for End-Users

Once a connection is established from J2EE to M via a connector proxy user, VistALink employs a process called *re-authentication*, which runs RPCs under the identity of an actual end-user.

Whenever an application uses a connection to execute a request, re-authentication makes a lightweight security context switch from the connector proxy user identity to actual end-user identities. This switch is performed for the following reasons:

- Connections are pooled for use by multiple end-users on the application server
- Most RPCs need to run in the context of specific Kernel/M end-users

NOTE: It is preferable that a connector proxy user have **no** privileges.

The VistALink re-authentication architecture assumes that the identity of the end-user has already been authenticated (verified) by the calling application. Therefore VistALink does not attempt to authenticate the end-user's identity. To do so would be difficult for an M system, given the variety of client mechanisms through which end-users could be accessing any J2EE application. (There is also the possibility that an end-user might not even have an account on a given M system – in this case, a special "application proxy" user account could be used instead.)

Instead of authenticating the end-user's identity, VistALink does the following to re-authenticate the end user:

1. Trusts that the connecting application has authenticated the end-user, relying on the chain of trust established through the connector proxy user
2. Verifies that it can match the end-user identity passed by the requesting application to an identity on the M system
3. Checks whether the identified M account is authorized to perform the requested action

When retrieving a VistALink connection from a connection factory, the application supplies end-user credentials as part of the connection specification. These credentials are used to switch security context on the M side. This re-authentication process establishes the correct end-user environment on the M server (VPID, Application Proxy, etc.) for the duration of the time that the connection is in use.

To link identities, the application selects one connection specification from the following, to retrieve a connection:

Connection Specification class	Credentials
VistaLinkDuzConnectionSpec	Known DUZ value, plus station number
VistaLinkVpidConnectionSpec	Known VPID value, plus station number
VistaLinkAppProxyConnectionSpec	Application Proxy name, plus station number

Table 8-1. Connection Specification Class and Credentials

VPID is the connection specification expected to be used in most production scenarios. Whatever end-user authentication mechanism is used by a Health*e*Vet-VistA application, the application should be able to obtain the VPID for a given end-user as an output from the authentication process. When the end-user is authenticated, the J2EE application can use the VPID as a way to identify the end-user to any VistA M systems, when executing RPCs on such systems on behalf of the end-user. Kernel patch XU*8.0*309 is required to support the VPID connection specification on M-VistA systems.

DUZ (with `DuzConnectionSpec`) is the primary re-authentication mechanism until the VPID infrastructure is fully rolled out. At that point `DuzConnectionSpec` will be deprecated, and `VpidConnectionSpec` will be the primary mechanism. In both cases (DUZ and VPID), it is expected that the login will have been performed already through a VHA-approved authentication mechanism, and that authentication mechanism will make the DUZ or VPID available for use by the application.

The Application Proxy connection specification is expected to be used in either of the following special situations:

- The J2EE end-user does not have a user account on the M system on which an RPC is to be executed
- It is not appropriate for the RPC to execute under the identity of a particular end-user.

8.5. RPC Security (B-Type Option)

All RPCs, whether accessed in J2SE client/server or J2EE mode, are secured with an RPC context (a "B"-type option). Any end-user for whom an RPC is executed must have the "B"-type option in their menu tree associated with the RPC. Otherwise an exception is thrown.

For more information on RPC security, see *Getting Started with the BDK, Chapter 3 (Extract): RPC Overview*, which is bundled in the /doc folder of the VistALink distribution, as the file **xwb1_1p13dg-rpc_extract.pdf.**

8.6. Creating Connector Proxy Users for J2EE Systems

To allow VistALink access to your M system from a specific J2EE system (application server), a Kernel "connector proxy" account must be used, as follows:

1. The M system manager creates a connector proxy account for a given J2EE system.
2. The M system manager provides the access and verify code of the connector proxy user to the J2EE system manager.
3. The J2EE system manager configures a connection pool to connect to the particular M system, using the access/verify code credentials of the connector proxy user provided by the M system manager.

8.6.1. Security Caution

By setting up connector proxy users, you are granting access on your M server to execute a *wide variety of RPCs* on your system. Therefore you need to do the following:

- **Create connection proxy users only for J2EE systems needing access to your M system.**
- **Give the access/verify codes of the connection proxy users to approved server administrators only.**

- **Create a different connection proxy user (with different access/verify code credentials) for each J2EE cluster (or data center) that will be connecting to your M system.**
- **Make sure not to disseminate the access/verify code for a connector proxy user outside of secure communication channels.**

8.6.2. Creating the "Connector Proxy User"

To create a connector proxy user:

1. You must hold the Kernel XUMGR key.
2. Add a new connector proxy user by using the Foundations menu on your M system and choosing the Enter/Edit Connector Proxy User option.
3. The account requires no additional information from what is prompted for by the option.
4. Leave the connector proxy user's Primary Menu empty.
5. Securely communicate the access code and verify code for the connector proxy user to the J2EE system manager setting up access from J2EE to your system. Also communicate the IP and port of your VistALink listener.

- **Do not enter divisions for a connector proxy user**
- **Do not enter a primary menu**
- **Do not also use the connector proxy user as a test "end-user"**
- **Utilize the user only as a connector proxy user**

8.7. Additional Security Issues (J2SE Mode)

The following issues pertain only to VistALink's client/server (J2SE) mode, where a Java client connects directly to an M system.

8.7.1. Authentication in Client/Server Mode

In client/server (J2SE) mode, VistALink authentication, like that of the RPC Broker, is a wrapper around the Kernel login on your M system. It obeys the same authentication rules as other Kernel logins on your system, and uses Kernel code to obtain a "yes/no" authentication decision for each login attempt.

As with the RPC Broker, authorization to execute RPCs is required, based on the "B"-type Kernel option/context mechanism.

8.7.2. CCOW-Based Single Sign-on

VistALink's client/server mode supports single sign-on to M systems based on user context, as implemented by the Office of Information's Single Sign-On/User Context (SSO/UC) project.

NOTE: On VMS-based systems, CCOW-based single sign on is dependent on the GETPEER^%ZOSV call to return the client IP address. The GETPEER^%ZOSV call is in turn dependent on the DCL COM file used to launch the VistALink listener. This file must set up logical symbols used by GETPEER^%ZOSV.

8.7.3. Kernel Auto Sign-on

VistALink's client/server (J2SE) supports an older single sign-on system, Kernel Auto-Sign on. In addition to the existing requirements for Kernel auto sign-on, VistALink has the following restrictions:

- The Broker client agent must already be running on the workstation.
- RPC Broker, Telnet, or VistALink can all be the first active connection.

NOTE: On VMS-based systems, Kernel auto sign-on for VistALink is dependent on the GETPEER^%ZOSV call to return the client IP address. The GETPEER^%ZOSV call is in turn dependent on the DCL COM file used to launch the VistALink listener. This file must set up logical symbols used by GETPEER^%ZOSV.

8.7.4. Sensitivity of VistALink-Logged Data

In client/server (J2SE) mode, VistALink uses the **log4J** logging utility to write information to a configurable log. The information written by VistALink loggers should not be application-sensitive data, and should not pose a security issue, even if the end-user turns on logging to its most detailed level.

9. Troubleshooting VistALink 1.6

9.1. Overview

In general, to troubleshoot most VistALink-related problems, you should check the following locations to gather helpful troubleshooting information:

- VistALink Administration Console – review the status and health indicators/failure counts for the connector in question, on the server in question.
- VistALink Administration Console – can you access the detail page for the connector, which makes a live query over the connector to the M system? Is the query successful? If not, what error message is displayed?
- Review the WebLogic server-specific log file and console output file for the server on which the error occurred:

```
<DOMAIN_HOME>/servers/<SERVER_NAME>/logs/<SERVER_NAME>.log
<DOMAIN_HOME>/servers/<SERVER_NAME>/logs/<SERVER_NAME>.out
```

- Review the log4j file logs configured for VistALink's logging, for the server on which the error occurred.
- Review the M-side error trap(s) for the M system(s) involved (D ^XTER)

9.2. Symptoms and Possible Solutions

The table below shows symptoms and possible causes and solutions for VistALink resource adapter problems. It is not an exhaustive list but contains some of the more common error conditions you may encounter.

9.2.1. Connection Attempt to Bad Listener Address/Port

Symptom: During WebLogic Server startup, an error like the following prints out on the command console and in the WebLogic Server log:

```
####<Jul 19, 2004 2:59:08 PM PDT> <Warning> <Connector> <testserver>
<myserver> <main> <<WLS Kernel>> <> <BEA-190032> <<
VistaLinkAdapter_vlj_testconnector > ResourceAllocationException of
gov.va.med.vistalink.adapter.cci.VistaLinkResourceException: Can not
create VistaSocketConnection;
    Root cause exception:
    gov.va.med.net.VistaSocketException: Can not create TCP/IP socket.;
    Root cause exception:
    java.net.ConnectException: Connection refused: connect on
    createManagedConnection.>
####<Jul 19, 2004 2:59:08 PM PDT> <Warning> <Connector> <testserver>
<myserver> <main> <<WLS Kernel>> <> <BEA-190024> <<
VistaLinkAdapter_vlj_testconnector > Error making initial connections for
```

```
pool. Reason: gov.va.med.net.VistaSocketException: Can not create TCP/IP
socket.;
   Root cause exception:
   java.net.ConnectException: Connection refused: connect>
```

Diagnoses and Solutions: The connector module's IP and port may be configured to connect to a non-existent listener.

9.2.2. Connection Attempt to RPC Broker Listener

Symptom: During WebLogic server start up, an error like the following prints out on the command console and in the WebLogic server log:

```
####<Jul 19, 2004 2:53:46 PM PDT> <Warning> <Connector> <testserver>
<myserver> <main> <<WLS Kernel>> <> <BEA-190032> <<
VistaLinkAdapter_vlj_testconnector > ResourceAllocationException of
gov.va.med.vistalink.adapter.cci.VistaLinkResourceException:
FoundationsException in executeInitSocketInteraction.;
   Root cause exception:
   gov.va.med.foundations.utilities.FoundationsException: Error executing
   the interaction;
   Root cause exception:
   gov.va.med.vistalink.adapter.spi.VistaLinkSocketClosedException: This
   connection cannot be retried because construction has failed;
   Root cause exception:
   gov.va.med.net.VistaSocketException: Error occurred reading from
   socket. ;
   Root cause exception:
   gov.va.med.net.VistaSocketException: End of stream encountered
   unexpectedly. on createManagedConnection.>
```

Diagnoses and Solutions: The connector module's IP and port may be configured to connect to an RPC Broker listener rather than to a VistALink listener.

9.2.3. Invalid Connector Proxy Credentials

Symptom: During WebLogic server start up, an error like the following prints out on the command console and in the WebLogic server log:

```
####<Jul 19, 2004 3:08:34 PM PDT> <Warning> <Connector> <testserver>
<myserver> <main> <<WLS Kernel>> <> <BEA-190032>
<< VistaLinkAdapter_vlj_testconnector > ResourceAllocationException of
gov.va.med.vistalink.adapter.cci.VistaLinkResourceException: Could not
perform logon[]Not a valid ACCESS CODE/VERIFY CODE pair. on
createManagedConnection.>

####<Jul 19, 2004 3:08:34 PM PDT> <Warning> <Connector> <testserver>
<myserver> <main> <<WLS Kernel>> <> <BEA-190024> <<
VistaLinkAdapter_vlj_testconnector > Error making initial connections for
```

```
pool. Reason: gov.va.med.vistalink.adapter.cci.VistaLinkResourceException:
Could not perform logon[]Not a valid ACCESS CODE/VERIFY CODE pair.>
```

Diagnoses and Solutions: The access and verify code specified for the connector are not valid connector proxy signon credentials on the destination M system.

9.2.4. Connector Proxy Expired Verify Code

Symptom: During WebLogic Server startup, errors like the following print out in the command console and the WebLogic Server log.

```
####<Jul 19, 2004 3:13:56 PM PDT> <Warning> <Connector> <testserver>
<myserver> <main> <<WLS Kernel>> <> <BEA-190032> <<
VistaLinkAdapter_vlj_testconnector > ResourceAllocationException of
gov.va.med.vistalink.adapter.cci.VistaLinkResourceException: Need new
Verify Code: true Need to select Divisions: false on
createManagedConnection.>

####<Jul 19, 2004 3:13:57 PM PDT> <Warning> <Connector> <testserver>
<myserver> <main> <<WLS Kernel>> <> <BEA-190024> <<
VistaLinkAdapter_vlj_testconnector > Error making initial connections for
pool. Reason: gov.va.med.vistalink.adapter.cci.VistaLinkResourceException:
Need new Verify Code: true Need to select Divisions: false>
```

Diagnoses and Solutions: The verify code specified for the connector has expired. Try setting **VERIFY CODE never expires?** to "Yes" for the connector proxy user. If this doesn't clear up the problem you may need to change the verify code of the connector proxy user and notify all connecting systems to update their connector configurations with the new verify code.

Note: All Connector Proxy users created using the Enter/Edit Connector Proxy User option (on the Foundations menu) should have **VERIFY CODE never expires?** automatically set to true.

9.2.5. Sporadic Socket Timeouts

Symptom: Application requests occasionally fail, and a VistaSocketTimeOutException is found in the log4j logs coincident with request failures. For example:

```
60871641 2008-03-11 15:07:37,520 [[ACTIVE] ExecuteThread: '1' for queue:
'weblogic.kernel.Default (self-tuning)'] DEBUG
gov.va.med.vistalink.adapter.spi.VistaLinkManagedConnection:executeInteraction:
1103 -
gov.va.med.vistalink.adapter.spi.VistaLinkManagedConnection[]10.6.15.10[]8016[]
1[]J2EE[fdi]1[mdi]5 M $JOB=24008
        Socket timeout has occurred
        gov.va.med.net.VistaSocketTimeOutException: Socket Timeout occured.
Timeout setting was: '15000 ms'. ;
        Root cause exception:
        java.net.SocketTimeoutException: Read timed out
java.net.SocketTimeoutException: Read timed out
        at java.net.SocketInputStream.socketRead0(Native Method)
        at java.net.SocketInputStream.read(SocketInputStream.java:129)
        at sun.nio.cs.StreamDecoder$CharsetSD.readBytes(StreamDecoder.java:411)
        at sun.nio.cs.StreamDecoder$CharsetSD.implRead(StreamDecoder.java:453)
        at sun.nio.cs.StreamDecoder.read(StreamDecoder.java:183)
        at java.io.InputStreamReader.read(InputStreamReader.java:167)
        at java.io.BufferedReader.fill(BufferedReader.java:136)
        at java.io.BufferedReader.read1(BufferedReader.java:187)
        at java.io.BufferedReader.read(BufferedReader.java:261)
        at java.io.Reader.read(Reader.java:122)
        at gov.va.med.net.SocketManager.receiveData(SocketManager.java:160)
        (etc.)
```

Diagnoses and Solutions: Requests from the J2EE side are occasionally timing out at the socket level. Response time may occasionally take longer than the configured connection timeout due to load on the target M system, network load, etc., particularly at the times of day with the heaviest system loads.

The partial log4j exception output above contains clues (the IP and port of the M system being connected to, and the timeout setting in use at the time of the socket timeout). If you are seeing socket timeout exceptions for a particular connector, you may want to consider raising the default timeout in the VistaLink connector configuration file for the adapter in question.

9.2.6. Connection Pool Capacity Exceeded

Symptom: Applications log the following error when trying to retrieve a connection from the connector module:

```
javax.resource.spi.ApplicationServerInternalException: Unable to get a
connection for VistaLinkAdapter_vlj_testconnector connection pool:
weblogic.common.resourcepool.ResourceLimitException: No resources
currently available in pool VistaLinkAdapter_vlj_testconnector to allocate
to applications, please increase the size of the pool and retry
```

Diagnoses and Solutions: There are no connections left in the pool to access. Check the size of the pool; you may need to increase it.

To monitor how many requests are being rejected, check the WebLogic console in the connector module in question. Look under the Monitoring tab at the Connections Rejected Total Count column.

Also, check the connection module for leaked connections: WebLogic console, resource adapter module, Monitoring tab, Number Detected Leaks column. If the application code does not close a connection, it can be leaked. Closing connections should usually be done in the "finally" portion of a try/catch/finally block.

9.2.7. ClassCastException

Symptom: `java.lang.ClassCastException` thrown to an application. When an application (such as the VistALink sample web application) tries to retrieve the VistALink connection factory from JNDI, and casts it to `VistaLinkConnectionFactory`, a `java.lang.ClassCastException` is thrown. A line of code like the following triggers the **ClassCastException**:

```
ic = new InitialContext();
VistaLinkConnectionFactory cf = (VistaLinkConnectionFactory)
   ic.lookup("myConnectorJndiName);
```

Diagnoses and Solutions: Usually this exception means that VistALink is not properly configured on the J2EE server. In particular, its libraries are not present on a classloader higher than that of the calling application. To get VistALink's libraries on a higher classloader, the two recommended approaches are:

<u>Production Systems</u>: Install the VistALink libraries as J2EE shared libraries.

<u>Non-production systems</u>: VistALink supporting libraries should be included in the exploded RAR, where they will be automatically loaded onto a high-level classloader by WebLogic.

Check you are using the appropriate method for your production or non-production server, and that the necessary supporting libraries are deployed (as J2EE shared libraries for production systems, or within the RAR for development systems):

- vljConnector-1.6.0.xxx.jar
- vljFoundationsLib-1.6.0.xxx.jar
- log4j-1.2.xx.jar

If you are still having difficulty, try turning on the following debug settings (WebLogic administration console, Server -> Debug tab) and then examining the server log output when the next ClassCastException is encountered:

- default.ClassFinder
- weblogic.classloader
- weblogic.classloader.Classloader
- weblogic.classloader.verbose
- weblogic.classloader.verbose.ClassLoaderVerbose

9.2.8. Configuration File Confusion

Symptom: The settings actually being used for deployed adapters (e.g., ip, and/or port, and/or JNDI name, etc.) don't correspond to the settings in my VistALink configuration file. Also, when I make changes in the settings in the configuration file, and redeploy my adapters, the settings actually used for the adapters don't change.

Diagnoses and Solutions: Check your classpath to see if more than one directory is included. If so, check to see if the file **gov.va.med.vistalink.connectorConfig.xml** is present in multiple directories on your classpath. If so, VistALink may be reading settings from the first VistALink configuration file that it finds on the classpath. In this case, remove all files of this name from directories on your server classpath, except for the one that should be the valid configuration file. Make sure the directory containing the valid VistALink configuration file is on your server classpath. Redeploy your connectors; the correct settings should now be read in for each adapter.

9.2.9. Reauthentication (End-User Identification) Failure

Symptom: Applications log the following error when trying to execute an RPC.

```
Exception:gov.va.med.vistalink.security.m.SecurityDuzDeterminationFaultExc
eption: Fault Code: 'Server'; Fault String: 'Internal Application Error';
Fault Actor: 'XOBV TEST PING'; Code: '182301'; Type: 'XOBV TEST PING';
Message: 'No valid DUZ found. [Security Type: DUZ] [DUZ Value:
'117075555']' at
gov.va.med.vistalink.rpc.RpcResponseFactory.handleSpecificFault(RpcRespons
eFactory.java:261) at … [deleted for brevity]
```

Diagnoses and Solutions: DUZ, the end-user credential used for re-authentication, is not valid on the system being connected to.

Note: The reported failure could also be “no valid VPID found”, or “no valid Application Proxy user found”, depending on the type of reauthentication used to identify the end-user.

9.2.10. Division Mismatch

Symptom: Applications log the following error when trying to execute an RPC:

```
Exception:gov.va.med.vistalink.security.m.SecurityDivisionDeterminationFau
ltException: Fault Code: 'Server'; Fault String: 'Internal Application
Error'; Fault Actor: 'XOBV TEST PING'; Code: '182302'; Type: 'XOBV TEST
PING'; Message: 'Division mismatch during user reauthentication for
security type [DUZ]. DUZ: [11707], Passed Division: [510] …… [deleted for
brevity]
```

Diagnoses and Solutions: The division passed in the end-user credential for re-authentication is not valid for the given user. See the sections of this guide *J2EE Re-Authentication Mechanisms for End Users* and *Institution Mapping.*

9.2.11. Failure to Authorize RPC (B-Type Option)

Symptom: Applications log the following error when trying to execute an RPC:

```
Exception:gov.va.med.vistalink.rpc.NoRpcContextFaultException: Fault Code:
'Server'; Fault String: 'Internal Application Error'; Fault Actor: 'XOBV
TEST PING'; Code: '182006'; Type: 'XOBV TEST PING'; Message: 'User
TESTER,JOE does not have access to option XOBV VISTALINK TESTER' at
gov.va.med.vistalink.rpc.RpcResponseFactory.handleSpecificFault(RpcRespons
eFactory.java:253) at … [excerpt]
```

Diagnoses and Solutions: The end-user lacks the "B"-type option necessary to execute the RPC in question. See the VistALink Developer’s Guide for information on writing RPCs.

9.2.12. M-Side Error Returned to J2EE

Symptom: Applications log the following error when trying to execute an RPC:

```
Exception occurred executing XOBV TEST PING
Exception:gov.va.med.vistalink.adapter.record.VistaLinkFaultException:
Fault Code: 'Server'; Fault String: 'System Error'; Fault Actor: ''; Code:
'181001'; Type: 'system'; Message: 'A system error occurred in M:
CALL+1^MYRPC' at
gov.va.med.vistalink.adapter.record.VistaLinkResponseFactoryImpl.handleFau
lt(VistaLinkResponseFactoryImpl.java:309) at … [excerpt]
```

Diagnoses and Solutions: An M error occurred, most likely in the RPC being executed. Check the M error log.

9.2.13. Abrupt Disconnects (M-Side Errors)

Symptom: <READ> or <ENDOFFILE> errors logged by the M listener.

Diagnoses and Solutions: On Caché systems, <READ> errors typically mean an abrupt disconnect between the WebLogic server and VistALink. For example, this could occur if the machine that the WebLogic server was running on was shut down without performing a graceful shutdown of the WebLogic server first.

<ENDOFFILE> errors may mean that the Caché server was shut down without shutting down the WebLogic server first (as often might be the case with enterprise-wide connections).

9.2.14. J2SE Connection Attempt to 1.0 Listener

Symptom: The following error message occurs when attempting to run the VistALink Swing Sample App

```
Could not create connection;
Root cause exception:
Gov.va.med.vistalink.adapter.cci.VistaLinkResourceException:RoundationsExce
ption executelnitSocket…
Root cause exception:
Gov.va.med.exception.FoundationsException: The response version [1.0] is
incompatible with this version...
```

Diagnoses and Solutions: The J2SE or J2EE client is attempting to connect to a VistALink/M server that is still running a VistALink 1.0 listener. Upgrade to the current VistALink 1.6 KIDS build.

9.3. Analyzing VistALink Errors in the M Error Trap

If you look through the M error trap, you may see errors with no "subject" line.
Such errors may have been logged by VistALink to note a fault condition detected by the listener. Unfortunately, at the present, there is no easy way to add a non-M fault to the error log and still have the subject show.

9.3.1. XOBSTR Variable

If you see an error log entry without a subject, enter XOBSTR at the "Which symbol? >" prompt. If the log entry is VistALink-related, that variable will be defined in the error trap, and its value should be an error message describing the fault condition encountered.

Example:

```
4)  13:08:13  VISTALINK,ROU  553187626  BG875
…
Which symbol? > XOBSTR

XOBSTR=Primary station &apos;&apos; of the request does not match the primary
station &apos;11000&apos; of the M/Kernel system.
```

Figure 9-1. M Error Trap

9.3.2. XWBTIP Variable

When examining a VistALink-related error trap, the XWBTIP variable may be useful in that it contains the IP address of the requesting site, as determined by the operating system:

```
1)  22:23:15  VL-HEAD,ROU    25537  |TCP|8016

Which symbol? > XWBTIP


XWBTIP=10.6.21.65
```

This variable may be useful in particular, when trying to determine whether a particular client system that is logging a large number of errors in the error trap, and if so, how to contact the managers of that system to correct the condition.

9.3.3. XOBLASTR Variable (M Request Timestamp)

There is a new variable that developers and sites can use for debugging:

- **XOBLASTR** Timestamp of last time any request was made on connection

Developers and site administrators can use this variable via JOBEXAM or the Cache´ console to watch activity and determine "dead" connections.

9.4. Analyzing VistALink Java Exceptions

For information about VistALink Java exceptions found in log file output, look up the exception class names in the VistALink Javadoc.

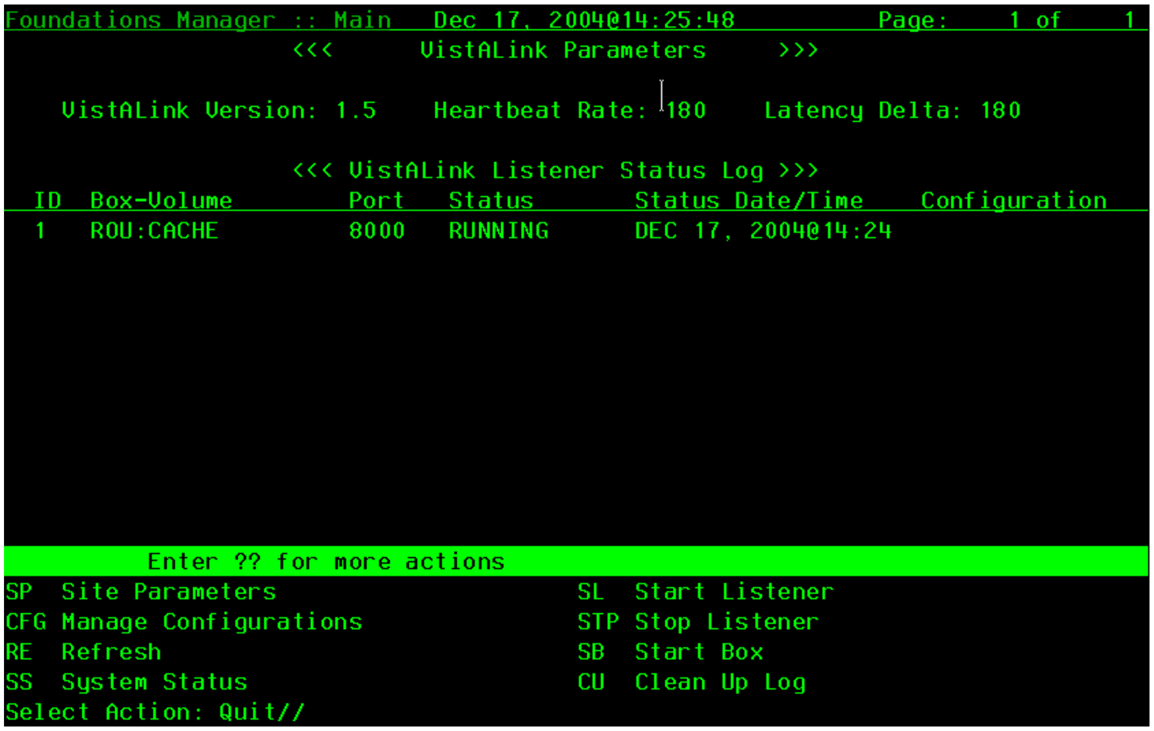
9.5. Section 508 Compliance: Differentiate Focus on Change Verify Code Check Box

Due to a known defect in Java Swing (http://bugs.sun.com/bugdatabase/view_bug.do?bug_id=4985801), and as defined by Section 508 compliance, it might be difficult for end-users to distinguish when focus occurs on the Change Verify Code check box in the login dialog window when using the Windows “Look and Feel” High Contrast Black color scheme.

Appendix A: Listener Management for Caché/NT Systems

For Caché/NT systems, listener processes are configured, started, and stopped entirely within the M environment. The Foundations Management menu [XOBU SITE SETUP MENU] is located under the Operations Management menu [XUSITEMGR] and provides several ListMan actions to do these operations. An example is shown in the figure below.

In Windows, unlike VMS-based systems, the VistALink listener runs as an M process. An application is provided to configure, start, and stop the listener in M-based VistA.



```
Foundations Manager :: Main   Dec 17, 2004@14:25:48         Page:    1 of    1
                    <<<      VistALink Parameters     >>>

    VistALink Version: 1.5    Heartbeat Rate: 180    Latency Delta: 180

                    <<< VistALink Listener Status Log >>>
  ID  Box-Volume         Port    Status        Status Date/Time    Configuration
  1   ROU:CACHE          8000    RUNNING       DEC 17, 2004@14:24









          Enter ?? for more actions
SP   Site Parameters                        SL   Start Listener
CFG Manage Configurations                   STP  Stop Listener
RE   Refresh                                SB   Start Box
SS   System Status                          CU   Clean Up Log
Select Action: Quit//
```

Figure A-1. Foundations Manager Interface (Cache´ Systems)

Creating/Editing Listener Configurations

To create or edit listener configuration entries, you should use the "Manage Configurations" action. The Manage Configurations action first prompts you to select a configuration entry. Then, within the entry, you can configure one or more listeners, as shown in the following example:

```
Select VistALink LISTENER CONFIGURATION NAME: DEFAULT

 NAME: DEFAULT// <Enter>
 Select PORT: 8000// <Enter>
 PORT: 8000// <Enter>
 STARTUP: YES// <Enter>
```

Figure A-2. Create or edit listener configuration entries

The table below defines the site parameter fields for a given listener entry:

Field	Meaning
Name	This field contains the name of the default VistALink configuration used with the associated BOX-VOLUME PAIR specified in the FOUNDATIONS SITE PARAMETERS file (#18.01).
Port	The port the listener will listen on.
Startup	If the listener should be started when this configuration is started, set this field to YES. Otherwise, set to NO. (If you want to keep the port in the configuration but temporarily do not want to start a listener, you would set this field to NO.)

Table A-1. Listener Configuration Entries Description Table

Starting All Configured Listeners

All listeners configured in the Foundations Site Parameters file for automatic startup, can be started with the Start Box action. This action will start all listeners for the configuration assigned to the current **BOX-VOLUME** pair.

For more information on configuring listeners for automatic startup, see the section of this chapter titled "Creating/Editing Listener Configurations."

Starting a Single Unconfigured Listener

To start a listener, use the Start Listener (SL) action on the Foundations Manager interface and enter the desired port. The action will first check if a listener is already listening on the port and inform you if it is. No damage to the system will occur if a listener is already listening on the port.

Stopping a Configured or Unconfigured Listener

To stop new connections:

1. Use the Stop Listener (STP) action and select the desired listener.

   This action may take up to 60 seconds to actually stop the listener. This prevents new connections from being established to the M system. It does not, however, terminate existing connections.

To terminate all connections and prevent new ones:

1. Use the Stop Listener (STP) action and select the desired listener. This action may take up to 60 seconds to actually stop the listener.

2. If you have access to the J2EE system (or can contact the system manager), STOP the associated VistALink connector. This shuts down the connection pool on the J2EE side and stops it from requesting new M connections.
3. You should shut down the Listener on the M system as well, to block any new connection requests that may come from other clients.

   NOTE: If the J2EE connector has not been stopped, it will continue to try to establish connections to the M system (which is why it is better to STOP the connector on the J2EE side).

4. Manually terminate any remaining XOBVSKT jobs running on the M system. This might include client/server connections or J2EE connectors that you were unable to stop.

Scheduling Listener Startup at System Startup

The XOBV LISTENER STARTUP option can be tasked to automatically start all required listener processes upon TaskMan start-up. This option starts all VistALink listener configuration operations at one time for the BOX-VOLUME pair. This type of start-up would be required after rebooting the system or restarting the configuration. After VistALink installation on a Cache´ system, this option should already be scheduled to run at startup.

To automatically start the listeners(s) when TaskMan is restarted, enter the XOBV LISTENER STARTUP option in the OPTION SCHEDULING file (#19.2). Schedule this option with SPECIAL QUEUING set to "Startup."

You can schedule the required options by making entries to the TaskMan Schedule/Unschedule Options, as shown in the figure below.

```
Select Systems Manager Menu Option: TASKMAN Management

Select Taskman Management Option: SCHedule/Unschedule Options

Select OPTION to schedule or reschedule: XOBV LISTENER STARTUP <Enter>  Start
VistALink Listener Configuration
         ...OK? Yes// <Enter>  (Yes)
                        Edit Option Schedule
    Option Name:   XOBV LISTENER STARTUP
    Menu Text:    Start VistALink Listener Configu      TASK ID:
_______________________________________________________________________________

  QUEUED TO RUN AT WHAT TIME:

DEVICE FOR QUEUED JOB OUTPUT:

 QUEUED TO RUN ON VOLUME SET:

      RESCHEDULING FREQUENCY:

             TASK PARAMETERS:

            SPECIAL QUEUEING:  STARTUP

_______________________________________________________________________________
```

Figure A-3. Automatically Starting Listener(s) on TaskMan Restart

Appendix B: Listener Management for DSM/VMS Systems

DSM/VMS systems should run VistALink as a TCP/IP service, commonly known as TCP/IP Service for OpenVMS (previously known as UCX). The TCP/IP Service permits multiple TCP/IP clients to connect and run as concurrent processes, up to the limits established by the system. TCP/IP listens on a particular port and launches a specified VistALink handler process for each client connection. Configuring, starting, and stopping listeners is managed entirely through the VMS TCP/IP Service.

DSM/VMS site systems are in the process of converting to Caché/VMS systems. **As such, the instructions below were written for the VistALink 1.0 release (November 21, 2003) and have not been modified for the conversion.**

Setting Up an OpenVMS User Account

The easiest way to configure an OpenVMS account to be a VistALink handler is to copy most of the parameters from VA MailMan TCP account. To do this:

1. Determine an unused User Identification Code (UIC), typically in the same group as other DSM for OpenVMS accounts.
2. Using the OpenVMS Authorize utility, copy the XMINET account to a new VLINK account with the unused UIC. You must have SYSPRV to do this.

Make sure that the account settings for the new VLINK account are the same as in the following example. If they are different, make sure that the impact of the different settings is acceptable for your system. In particular, make sure that the DisCtlY, Restricted, and Captive flags are set for security reasons.

Setting Up a Home Directory for the VistALink Handler Account

You need to create a home directory for the VistALink handler account. This directory will house the DCL command procedure that is executed whenever a client connects, as well as log files. Make sure that the owner of the directory is the VLINK account.

For example, to create a home directory named [VLINK] with ownership of VLINK, enter:

```
$ CREATE/DIR [VLINK]/OWNER=VLINK
```

Creating a DCL Login Command Procedure for the VistALink Handler

To create a DCL login procedure for the VistALink Handler:

1. Use the home directory for the handler account.
2. Make sure the command procedure file is owned by the VistALink handler account.
3. Adjust the DSM command line (environment, UCI, and volume set) for your system.
4. If access control is enabled, ensure that the VLINK account has access to this UCI, volume set, and routine. (See "Access Control List (ACL) Issues", later in this chapter).

It is possible to run different TCP/IP Service listener processes on multiple nodes.

Sample DCL Login Command Procedure

In the sample procedure below, ‘environ,’ ‘uci,’ and ‘vol’ are site-specific.

```
$!VLINK.COM - for incoming connect requests
$!--------------------------------------------------------------
$set noon        !Don't stop
$ set noverify    !change as needed
$! set verify     !change as needed
$! WAIT 00:00:00
$ purge/keep=10 sys$login:VLINK*.log !Purge log files only
$! set proc/priv=(share)  !May not be required for MBX device
$  x=f$trnlnm("sys$net")     !This is our MBX device
$
$ write sys$output "Opening "+x !This can be viewed in the log file
$! Check status of the BG device before going to DSM
$ cnt=0
$ CHECK:
$ stat=f$getdvi("''x'","STS")
$ if cnt .eq. 10
$ then
$ write sys$output "Could not open "+ x
$ goto EXIT
$ else
$       if stat .ne. 16
$       then
$       cnt = cnt + 1
$       write sys$output "''cnt'> ''x' not ready!"
$       wait 00:00:01 !Wait one second to assure connection
$       goto CHECK
$       else
$       SET NOVERIFY
$!----------------------------------------------------------------
$       dsm/env=DSMMANAG /uci=VAH  /vol=ROU UCX^XOBVTCP
$!----------------------------------------------------------------
$       endif
$ endif
$ EXIT:
$ logout/FULL
```

Figure B-1. Sample DCL Login Command Procedure

Setting Up and Enabling the TCP/IP Service

Once you create the VistALink handler, create the TCP/IP Service to listen for connections and launch the VistALink handler. You need to choose:

- The OpenVMS node to run the listener on.

  Choose the node that you want to run the resulting M jobs on to process incoming VistALink messages. This is also the node whose IP address will be advertised to other systems as the location of your VistALink listener.

- The port it should listen on.

- The user account and command file name to invoke when a connection is received.

The steps to set up a TCP/IP Service for VistALink are:

1. Determine the port
2. Set up the "VLINK" TCP/IP Service
3. Enable and save the "VLINK" TCP/IP Service

Prior to setting up TCP/IP in production, you can set up a "test" TCP/IP Service that logs into an M test account. The test TCP/IP Service can use the same OpenVMS account and directory as the production TCP/IP Service. Just create a different DCL command file using the UCI and volume set of the test account.

Obtaining an Available Listener Port (for Alpha/VMS systems only)

Port selections conflict only if another process is using the same port on the same system. To list the ports currently in use on OpenVMS systems, use the DCL command:

```
$  TCPIP SHOW DEVICE_SOCKET
   Port                Remote
Device_socket  Type    Local  Remote  Service          Host

  bg3         STREAM    8001       0  VistALink        0.0.0.0
  bg23        STREAM    9700       0  Z3ZTEST          0.0.0.0
  bg24        STREAM    9600       0  ZSDPROTO         0.0.0.0
```

Figure B-2. Obtaining an available listener port (for Alpha/VMS systems only)

If ‘8000’ appears in the Local Port column, another application is already using this port number; so you should choose another port.

Creating the TCP/IP Service

Since the TCP/IP Service is node specific, make sure you are on the same node that you want the listener to run on. Follow this example to create the service:

```
$TCPIP
TCPIP> SET SERVICE VLINK/USER=VLINK/PROC=VLINK /PORT=8000-
_TCPIP> /PROTOCOL=TCP/REJECT=MESSAGE="All channels busy" -
_TCPIP> /LIMIT=50/FILE=SYS$SYSDEVICE:[VLINK]VLINK.COM

TCPIP> SHO SERVICE VLINK/FULL

Service: VLINK
                           State:     Disabled
Port:             8000     Protocol:  TCP            Address:  0.0.0.0
                           User_name: not defined    Process:  VLINK
```

Figure B-3. Creating the TCP/IP Service

Enabling and Saving the Service

Because the TCP/IP service is node specific, make sure you are on the same node that you want the listener to run on. Follow this example to enable the service:

```
TCPIP> ENABLE SERVICE VLINK          (enable service immediately)
TCPIP> SET CONFIG ENABLE SERVICE VLINK  (save service for reboot)
TCPIP> SHO SERVICE/FULL VLINK

Service: VLINK
                           State:     Enabled
Port:            8000     Protocol:  TCP             Address:  0.0.0.0
Inactivity:         5     User_name: VLINK         Process:  VLINK
Limit:             50     Active:      0             Peak:       0

File:         SYS$SYSDEVICE:[VLINK]VLINK.COM
Flags:        Listen

Socket Opts:  Rcheck Scheck
 Receive:           0     Send:               0

Log Opts:     None
 File:        not defined

Security
 Reject msg:  All channels busy

 Accept host: 0.0.0.0
 Accept netw: 0.0.0.0
TCPIP> SHO CONFIG ENABLE SERVICE

Enable service
     FTP, FTP_CLIENT, VLINK, MPI, TELNET, XMINETMM
TCPIP> EXIT
```

Figure B-4. Enabling and saving the TCP/IP service

NOTE: To test the connection, refer to the section of this guide called "Testing the Listener."

Access Control List ACL Issues

Some sites use DSM's ACL feature, which controls access explicitly to each OpenVMS account needing to enter that DSM environment. If your site is using ACL, you should set up the VLINK account with PROGRAMMER access, and then specify the Volume set and UCI name that the VLINK user account has authorization to access. Ensure that the OpenVMS VLINK account allows only Network longings, and prohibits Batch, Local, Dialup and Remote logins.

The following is an example of setting this level of access for a VLINK account:

```
$ DSM /MAN ^ACL

Environment Access Utilities

    1.  ADD/MODIFY USER                   (ADD^ACL)
    2.  DELETE USER                       (DELETE^ACL)
    3.  MODIFY ACTIVE AUTHORIZATIONS      (^ACLSET)
    4.  PRINT AUTHORIZED USERS            (PRINT^ACL)

Select Option > 1  ADD/MODIFY USER

OpenVMS User Name:   >   VLINK

ACCESS MODE    VOL       UCI       ROUTINE
-----------    ---       ---       -------

No access rights for this user.

Access Mode ([M]ANAGER, [P]ROGRAMMER, or [A]PPLICATION):   >   P
Volume set name:   >   ROU
UCI:   >   VAH
UCI:   >   <RET>
Volume set name:   >   <RET>
Access Mode ([M]ANAGER, [P]ROGRAMMER, or [A]PPLICATION):   >   <RET>

USER           ACCESS MODE    VOL       UCI       ROUTINE
----           -----------    ---       ---       -------

VLINK        PROGRAMMER     ROU       VAH

OK to file?   <Y>   <RET>

OpenVMS User Name:   >   <RET>

OK to activate changes now?   <Y>   <RET>

Creating access authorization file:  SYS$SYSDEVICE:[DSMMGR]DSM$ACCESS.DAT.
```

Figure B-5. Access Control List

Controlling the Number of Log Files

The VLINK TCP/IP Service automatically creates log files in the VLINK directory named "VLINK.LOG;xxx," where 'xxx' is a file version number. This cannot be prevented. New versions of this file will be created until that file version number reaches the maximum number of 32767. To minimize the number of log files created, you may want to initially rename this log file to the highest version number with the command:

```
$ RENAME disk$:[VLINK]VLINK.LOG; disk$:[VLINK]VLINK.LOG;32767
```

Alternatively, you can set a limit on the number of versions of the log file that can concurrently exist in the VLINK directory:

```
$ SET FILE /VERSION_LIMIT=10 disk$:[VLINK]VLINK.LOG;
```

NOTE: You probably should not limit the number of versions of the log file until you know your VistALink service is working correctly. Keeping the log files can help when diagnosing problems with the service/account.

Disabling New Connections

1. To stop incoming connections, stop the TCP/IP service.

   This prevents new connections from being established to the M system. Current connections (that have their own connection, PID, etc) will remain, but new connections cannot be made until the service is enabled again.

Terminating All Connections and Preventing New Ones

1. To stop incoming connections, stop the TCP/IP service.
2. If you have access to the J2EE system (or can contact the system manager), disable the associated VistALink connector. This shuts down the connection pool on the J2EE side and stops it from requesting new M connections. You should shut down the Listener on the M system as well, to block any new connection requests that may come from other clients.

   NOTE: If the J2EE connector has not been stopped, however, it will continue to try to establish connections to the M system (which is why it is better to STOP the connector on the J2EE side).

3. Manually terminate any remaining XOBVSKT jobs running on the M system. This might include client/server connections, or J2EE connectors that you were unable to stop.

Glossary

AAC	Formerly the Austin Automation Center. Renamed to the Austin Information Technology Center (AITC)
Access Code	A password used by the Kernel system to identify the user. It is used with the verify code.
Adapter	Another term for *resource adapter* or *connector*.
Administration Server	Each Oracle WebLogic server domain must have one server instance that acts as the administration server. This server is used to configure all other server instances in the domain.
AITC	*Austin Information Technology Center*
Alias	An alternative filename.
Alpha/VMS	Alpha: Hewlett Packard computer system VMS: *Virtual Memory System*
Anonymous Software Directories	M directories where VHA application and patch zip files are placed for distribution.
API	*Application Program Interface*
Application Proxy User	A Kernel user account designed for use by an application rather than an end-user.
Application Server	Software/hardware for handling complex interactions between users, business logic, and databases in transaction-based, multi-tier applications. Application servers, also known as app servers, provide increased availability and higher performance.
ASTM	*American Society for Testing and Materials*
Authentication	Verifying the identity of the end-user.
Authorization	Granting or denying user access or permission to perform a function.
Base Adapter	Version 8.1 of WebLogic introduced a "link-ref" mechanism enabling the resources of a single "base" adapter to be shared by one or more "linked" adapters. The base adapter is a standalone adapter that is completely set up. Its resources (classes, jars, etc.) can be linked to and reused by other resource adapters (linked adapters). The deployer only needs to modify a subset of the linked adapters' deployment descriptor settings. *Note: This mechanism is no longer supported in WebLogic 9 and later for J2CA 1.5 adapters (e.g., VistALink 1.6).*
BEA WebLogic	BEA WebLogic is a J2EE Platform application server. Oracle has acquired BEA Systems, Inc. From here forward it will be referred to as Oracle.
Caché	Caché is an M environment, a product of InterSystems Corp.
Cache/VMS	Cache: InterSystems Caché object database that runs SQL VMS: *Virtual Memory System*
CCI	Common Client Interface

CCOW	A standard defining the use of a technique called "context management," providing the clinician with a unified view on information held in separate and disparate healthcare applications that refer to the same patient, encounter or user.
Classpath	The path searched by the JVM for class definitions. The class path may be set by a command-line argument to the JVM or via an environment variable.
Client	Can refer to both the client workstation and the client portion of the program running on the workstation.
Connection Factory	A J2CA class for creating connections on request.
Connection Pool	A cached store of connection objects that can be available on demand and reused, increasing performance and scalability. VistALink uses connection pooling when running on a J2EE server.
Connector	A system-level driver that integrates J2EE application servers with Enterprise Information Systems (EIS). VistALink is a J2EE connector module designed to connect to Java applications with VistA/M systems. The term is used interchangeably with *connector module*, adapter, *adapter module*, and *resource adapter*.
Connector Proxy User	For security purposes, each instance of a J2EE connector must be granted access to the M server it connects to. This is done via a Kernel user account set up on the M system. This provides initial authentication for the app server and establishes a trusted connection. The M system manager must set up the connector user account and communicate the access code, verify code and listener IP address and port to the J2EE system manager.
COTS	*Commercial, Off-The-Shelf*
CSV	*Comma-Separated Values* format
DBF	Database file format underlying many database applications (originally dBase)
DCL	*Digital Command Language.* An interactive command and scripting language for VMS.
Division	VHA sites are also called *institutions*. Each institution has a *station number* associated with it. Occasionally a single institution is made up of multiple sites, known as *divisions*. To make a connection, VistALink needs a station number from the end-user's New Person entry in the KERNEL SYSTEM PARAMETERS file (#8989.3). It looks first for a division station number and if it can't find one, uses the station number associated with default institution.
DSM	*Digital Standard MUMPS.* An M environment, a product of InterSystems Corp.
DUZ	A local variable holding a number that identifies the signed-on user. The number is the Internal Entry Number (IEN) of the user's record in the NEW PERSON file (file #200)
EAR file	*Enterprise archive* file. An enterprise application archive file that contains a J2EE application.
EIS	*Enterprise Information System*
EPHI	*Electronic Protected Health Information*

FatKAAT	*Fat-Client (i.e. Rich client) Kernel Authentication and Authorization*
FDA	*FileMan Data Array*
File #18	SYSTEM file #18 was the precursor to the KERNEL SYSTEM PARAMETERS file (#8989.3), and is now obsolete. It uses the same number space that is now assigned to VistALink. Therefore, File #18 must be deleted before VistALink can be installed.
Global	A multi-dimensional data storage structure -- the mechanism for persistent data storage in a MUMPS database.
HealtheVet-VistA	The VHA is converting its MUMPS-based VistA healthcare system to a new J2EE-based platform and application suite. The new system is known as HealtheVet-VistA.
HIPAA	*Health Insurance Portability and Accountability Act*
IDE	*Integrated development environment.* A suite of software tools to support writing software.
Institution	VHA sites are also called *institutions*. Each institution has a *station number* associated with it. Occasionally a single institution is made up of multiple sites, known as *divisions*. To make a connection, VistALink needs a station number from the end-user's New Person entry in the KERNEL SYSTEM PARAMETERS file (#8989.3). It looks first for a division station number and if it can't find one, uses the station number associated with default institution.
Institution Mapping	The VistALink 1.6 release includes a small utility that administrators can use to associate station numbers with JNDI names, and which allows runtime code to retrieve the a VistALink connection factory based on station number.
ISO	*Information Security Officer*
J2CA	J2EE Connector Architecture 1.6
J2CA	*J2EE Connector Architecture*. J2CA is a framework for integrating J2EE-compliant application servers with Enterprise Information Systems, such as the VHA's VistA/M systems. It is the framework for J2EE connector modules that plug into J2EE application servers, such as the VistALink adapter.
J2EE	The *Java 2 Platform, Enterprise Edition (J2EE)* is an environment for developing and deploying enterprise applications. The J2EE platform consists of a set of services, APIs, and protocols that provide the functionality for developing multi-tiered, Web-based applications. A J2EE Connector Architecture specification for building adapters to connect J2EE systems to non-J2EE enterprise information systems.
J2EE	*Java 2 Enterprise Edition.* A standard suite of technologies for developing distributed, multi-tier, enterprise applications.
J2SE	*Java 2 Standard Edition.* Sun Microsystem's programming platform based on the Java programming language. It is the blueprint for building Java applications, and includes the Java Development Kit (JDK) and Java Runtime Environment (JRE).
JAAS	*Java Authentication and Authorization Service.* JAAS is a pluggable Java framework for user authentication and authorization, enabling services to

	authenticate and enforce access controls upon users.
JAR file	*Java archive* file. It is a file format based on the ZIP file format, used to aggregate many files into one.
Java Library	A library of Java classes usually distributed in JAR format.
Javadoc	Javadoc is a tool for generating API documentation in HTML format from doc comments in source code. Documentation produced with this tool is typically called Javadoc.
JBoss	JBoss is a free software / open source Java EE-based application server.
JCA CCI	*J2EE Connector Architecture Common Client Interface*
JDK	*Java Development Kit*. A set of programming tools for developing Java applications.
JMX	*Java Management eXtensions*. A java specification for building manageability into java applications, including J2EE-based ones.
JNDI	*Java Naming and Directory Interface*. A protocol to a set of APIs for multiple naming and directory services.
JRE	The *Java Runtime Environment* consists of the Java virtual machine, the Java platform core classes, and supporting files. JRE is bundled with the JDK but also available packaged separately.
JSP	*Java Server Pages*. A language for building web interfaces for interacting with web applications.
JVM	*Java Virtual Machine*. The JVM interprets compiled Java binary code (byte code) for specific computer hardware.
KAAJEE	*Kernel Authentication and Authorization for Java 2 Enterprise Edition*
Kernel	Kernel functions as an intermediary between the host M operating system and VistA M applications. It consists of a standard user and program interface and a set of utilities for performing basic VA computer system tasks, e.g., Menu Manager, Task Manager, Device Handler, and security.
KIDS	*Kernel Installation and Distribution System*. The VistA/M module for exporting new VistA software packages.
LDAP	Acronym for *Lightweight Directory Access Protocol*. LDAP is an open protocol that permits applications running on various platforms to access information from directories hosted by any type of server.
Linked Adapter	Version 8.1 of WebLogic introduced a "link-ref" mechanism enabling the resources of a single "base" adapter to be shared by one or more "linked" adapters. The base adapter is a standalone adapter that is completely set up. Its resources (classes, jars, etc.) can be linked to and reused by other resource adapters (linked adapters). The deployer only needs to modify a subset of linked adapters' deployment descriptor settings. *Note: This mechanism is no longer supported in WebLogic 9 and later for J2CA 1.5 adapters (e.g., VistALink 1.6).*
Linux	An open-source operating system that runs on various types of hardware platforms. HealtheVet-VistA servers use both Linux and Windows operating systems.

Listener	A socket routine that runs continuously at a specified port to field incoming requests. It sends requests to a front controller for processing. The controller returns its response to the client through the same port. The listener creates a separate thread for each request, so it can accept and forward requests from multiple clients concurrently.
log4J Utility	An open-source logging package distributed under the Apache Software license. Reviewing log files produced at runtime can be helpful in debugging and troubleshooting.
logger	In log4j, a logger is a named entry in a hierarchy of loggers. The names in the hierarchy typically follow Java package naming conventions. Application code can select a particular logger by name to write output to, and administrators can configure where a particular named logger's output is sent.
M (MUMPS)	*Massachusetts General Hospital Utility Multi-programming System*, abbreviated M. M is a high-level procedural programming computer language, especially helpful for manipulating textual data.
Managed Server	A server instance in a Oracle WebLogic domain that is not an administration server, i.e., not used to configure all other server instances in the domain.
MBeans	In the Java programming language, an MBean (managed bean) is a Java object that represents a manageable resource, such as an application, a service, a component, or a device. MBeans must be concrete Java classes.
Messaging	A framework for one application to asynchronously deliver data to another application, typically using a queuing mechanism.
Multiple	A VA FileMan data type that allows more than one value for a single entry.
Namespace	A unique 2-4 character prefix for each VistA package. The DBA assigns this character string for developers to use in naming a package's routines, options, and other elements. The namespace includes a *number space*, a pre-defined range of numbers that package files must stay within.
New Person File	The New Person file contains information for all valid users on an M system.
NIST	*National Institute for Standards and Technology*
OI	*Office of Information*
Oracle WebLogic	Oracle WebLogic is a J2EE Platform application server. Oracle has acquired BEA Systems, Inc.
OS	*Operating System*
Patch	An update to a VistA software package that contains an enhancement or bug fix. Patches can include code updates, documentation updates, and information updates. Patches are applied to the programs on M systems by IRM services.
PHI	*Protected Health Information*
ra.xml	ra.xml is the standard J2EE deployment descriptor for J2CA connectors. It describes connector-related attributes and its deployment properties using a standard DTD (Document Type Definition) from Sun.
Re-authentication	When using a J2CA connector, the process of switching the security context of the connector from the original application connector "user" to the actual end-

	user. This is done by the calling application supplying a proper set of user credentials.
Resource Adapter	J2EE resource adapter modules are system-level drivers that integrate J2EE application servers with Enterprise Information Systems (EIS). This term is used interchangeably with *resource adapter* and *connector*.
RM	*Requirements Management*
Routine	A program or sequence of computer instructions that may have some general or frequent use. M routines are groups of program lines that are saved, loaded, and called as a single unit with a specific name.
RPC	*Remote Procedure Call*. A defined call to M code that runs on an M server. A client application, through the RPC Broker, can make a call to the M server and execute an RPC on the M server. Through this mechanism a client application can send data to an M server, execute code on an M server, or retrieve data from an M server
RPC Broker	The RPC Broker is a client/server system within VistA. It establishes a common and consistent framework for client-server applications to communicate and exchange data with VistA/M servers.
RPC Security	All RPCs are secured with an RPC context (a "B"-type option). An end-user executing an RPC must have the "B"-type option associated with the RPC in the user's menu tree. Otherwise an exception is thrown.
SAD	*Software Architecture Document*
SE&I	*Software Engineering & Integration*
Servlet	A Java program that resides on a server and executes requests from client web pages.
Socket	An operating system object that connects application requests to network protocols.
SPI	*J2CA service provider interface* Service-Level Contract
SRS	*Software Requirements Specification*
SSH	[1]Secure Shell or SSH is a network protocol that allows data to be exchanged using a secure channel between two networked devices. Used primarily on Linux and Unix based systems to access shell accounts, SSH was designed as a replacement for Telnet and other insecure remote shells, which send information, notably passwords, in plaintext, leaving them open for interception. The encryption used by SSH provides confidentiality and integrity of data over an insecure network, such as the Internet.
TCP/IP	*Transmission Control Protocol (TCP) and the Internet Protocol (IP)*
Term	Definition
TXT	*Text* file format
UML	*Unified Modeling Language* is a standardized specification language for object modeling.

[1] http://en.wikipedia.org/wiki/Secure_Shell

VA	*Department of Veterans Affairs*
VACO	*Veterans Affairs Central Office*
Verify Code	A password used in tandem with the access code to provide secure user access. The Kernel's Sign-on/Security system uses the verify code to validate the user's identity.
VistA	*Veterans Health Information Systems and Technology Architecture*. The VHA's portfolio of M-based application software used by all VA medical centers and associated facilities.
VistALink Libraries	Classes written specifically for VistALink.
VL	*VistaLink* is a runtime and development tool providing connection and data conversion between Java and M applications in client-server and n-tier architectures, to which this document describes the architecture and design.
VMS	*Virtual Memory System*. An operating system, originally designed by DEC (now owned by Hewlett-Packard), that operates on the VAX and Alpha architectures.
VPID	*VA Person Identifier*. A new enterprise-level identifier uniquely identifying VA 'persons' across the entire VA domain.
WAR file	*Web archive* file. Contains the class files for servlets and JSPs.
WebLogic Server	A J2EE application server manufactured by Oracle WebLogic Systems.
WebSphere	WebSphere Application Server (WAS) is and IBM application server.
XLS	Microsoft Office XL worksheet and workbook file format
XML	*Extensible Markup Language*
XmlBeans	XMLBeans is a Java-to-XML binding framework which is part of the Apache Software Foundation XML project.
XOB Namespace	The VistALink namespace. All VistALink programs and their elements begin with the characters "XOB."

REF: For a comprehensive list of commonly used infrastructure- and security-related terms and definitions, please visit the Security and Other Common Services Glossary Web page at the following Web address:

http://vista.med.va.gov/iss/glossary.asp

For a comprehensive list of acronyms, please visit the Security and Other Common Services Acronyms Web site at the following Web address:

http://vista/med/va/gov/iss/acronyms/index.asp