MUSE

Installation Manual

31 March 2014

Document Version 5.9.1.3
Muse 2.7.0.0

Notice

Disclaimer

Trademarks

www.museglobal.com

One Embarcadero, Suite 500
SAN FRANCISCO, CA 94111 U.S.A.
415-896-6873 FX 415-896-6870 SUPPORT@MUSEGLOBAL.COM

Table of Contents

1.0

Product Overview

Muse is a product for providing search services to users, allowing them to search across multiple databases simultaneously irrespective of the protocol of the database search engines, and to manually or automatically manipulate the retrieved record sets to remove duplicates and unwanted records.

Muse is web based, and interacts with the user through a standard web browser. An organization may maintain many different Muse interfaces for use by individuals or groups of users so users may interact in the language of their choice and with the degree of complexity and help they desire.

Muse allows the organization to select a number of databases and search engines for particular users, which can be searched individually or with a single search statement. Protocols and indexing schemes are translated automatically so the target system is given the user's query in a form it understands. The results of searches are presented to the user as single result sets. These can be condensed or merged, or the desired manipulation can be defined in advance, so a single combined set of unique results from all searched databases is presented immediately, ranked according to the user's criteria.

Muse can deliver full objects (such as full text documents, images, video clips, etc.) once the user has selected those of interest, and includes various mechanisms for delivery. Comprehensive rights management is provided, so costs associated with various online and offline activities can be correctly accounted for and apportioned.

Muse is a scalable system, able to operate across the Internet or a local area network to utilize available computing power as the needs of users grow. It has an extensible architecture where the Processing Modules are loaded as needed for efficiency, and the structure of the system can easily utilize specialist processing from external programs. The system can be searched through the interface provided by Muse, or Muse can be built into another system via a program interface to provide enhanced services.

2.0

System Requirements and Preparation

Muse can run on any platform on which a Java Virtual Machine can be installed. These system re quirements and the subsequent installation process assume the installation of a complete Muse system. If less than the complete system is to be installed (for example, if Muse will be installed behind the scenes as part of an existing front-end application) then not all the described components are required. However, this document outlines the maximum requirements.

Note: The "Muse Server Sizing.pdf" document can also be used as a guide to set up a Muse Search environment. It contains information about hardware and network specifications.

2.1 Hardware

The following are the minimum and recommended hardware specifications for running a Muse server. For this purpose we consider the Muse server will be capable of handling a minimum of 10 sim ultaneously connected users (simultaneous sessions) with a quick response time.

Minimum specification:

- 50 GB hard disk space
- 2 GB RAM
- Intel Pentium IV (or equivalent)

Recommended specification:

- 80 GB hard disk space
- 4 GB RAM
- Intel Pentium IV (or equivalent) or better processor / multiprocessors
- Processor clock speed of 3 GHz or better (for Pentium and equivalent processors)

2.2 Software

The following environment software products are necessary to operate Muse. Where a suitable operating system is in use, it is not necessary to have a dedicated computer to run Muse. It can co-exist with other application software on a single computer running a suitable operating system.

All the environment software listed below has been tested with Muse and supports it fully. This does not imply that Muse is not fully functional on other environments not listed here. If the environment you need for your Muse installation is not listed here, please contact your Muse provider for further details.

- Operating system

  The following operating systems have been tested with Muse:

  - Windows 2000, Windows XP, Windows Server 2003, Windows 7, Windows Server 2008
  - Sun Solaris (x86 and SPARC), v. 8-10
  - Mac OS X: 10.0-10.3, 10.4 PPC
  - Linux (x86) - various flavors: RedHat, Fedora, CentOS, Debian, SuSE, Mandriva, FreeBSD, Slackware, Gentoo

    Note: The only shells that are currently supported are: csh, bash (for UNIX environments) and windows command prompt (for Microsoft Windows environments). The provided UNIX scripts may also work with tcsh, ash. However, these have not been not tested and may not work as expected.

- Network settings

  The computer running Muse must have a properly configured Netwok Interface Card. The system configuration file which associates an IP with a name must be well configured. For example, on a Solaris system, the file `/etc/hosts` should contain pairs of the form `"IP name"`. A mandatory pair is the one that associates the `localhost` to an IP `"127.0.0.1 localhost"`.

  If the server has an address obtained through DHCP, then every such possible IP must be associated with the respective Serial Number and registered in Muse or else the IP protection will prevent the ICE server from starting. If not all the potential IPs are registered, after a server restart a different IP may be set to the server, which is not registered in the Registration service. This would be just killing as it would require a Registration Extension every time, with the whole procedure of approval on MuseGlobal side and re-run the Muse Registration setup on client side. Also, the show stopper in case of DHCP is the fact that one (either end-user, or client program)

could not access the Muse system, from the Intranet/Internet, because its IP may change with every restart (or DHCP timeout), and hence it is not known a priori – it cannot be a service in other words.

Thus, the computer must have a static IP and if it is obtained through DHCP, then that IP must be every time the same. Also, it should have a Fully Qualified Domain name attached, and if it is going to be used over the Internet, the IP address must be public.

- JDK (Java Development Kit). The installation of a SDK or JDK package is a requirement since they contain tools not available in the other packages that can be used for monitoring and troubleshooting issues related to the Java Virtual Machine and the Muse software. For example, the `jstat` tool available in a JDK package displays performance statistics for an instrumented HotSpot Java virtual machine (JVM). The Java Virtual Machine Software Development Kit(JDK) contains the JRE.

The following virtual machines and platform combinations have been tested with Muse:

- Intel x86/Win32 (NT/2000/XP), JDK 1.6
- Intel 64/Win64(XP/2008), JDK 1.6
- Intel x86/Solaris, JDK 1.6
- Intel x86/Linux, JDK 1.6
- Intel 64/Linux 64, JDK 1.6
- Intel x86/Linux, JDK 7

Please note the following known issues with different Java versions on certain operating systems:

There is a problem running Muse on JRE 1.6 rc1. This problem seems to be related to the XSLT Processor used by default by the JVM. Specifying the processor as a system property seems to solve this problem. However, our recommendation is not to use this version of the JVM.

Note: It is recommended to have installed the latest build of JDK 1.6 available. We do not recommend JDK versions smaller than 1.6 Update 18.

To have internationalization support in Muse the JDK must include support for all languages. OracleJVM for Windows platforms currently offer two versions, US-only and international, so the `Windows (all languages, including English)` option must be selected. This is not a concern for other operating systems, as required support for internationalization is always present.

If you do not have a Java Virtual Machine, you can obtain one from one of the following vendors:

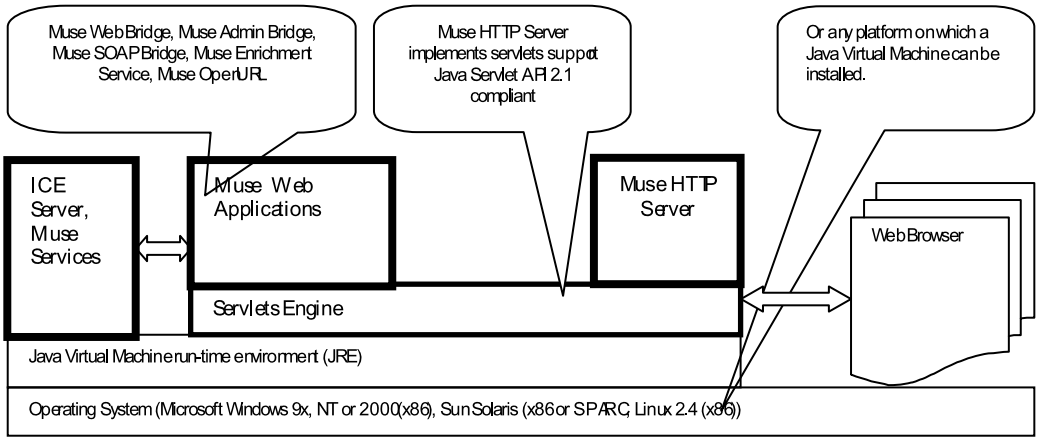
Vendor	URL Address
Sun	http://www.oracle.com/technetwork/java/javase/downloads/jdk6-downloads-1637591.html
IBM	http://www.ibm.com/developerworks/java/jdk/

Note: There is no IBM JDK for Windows platform.

Note: You must obtain the Java Virtual Machine only from SUN or IBM. Muse works if and only if the Java Virtual Machine is from Sun or IBM. Note that there is no IBM JDK for Windows platform.

Conforming to the client-server architectural model, the Muse environment provides servlets support for various Web applications requiring an HTTP Web server. The Muse package incorporates the Apache Tomcat Server.

Software environment architecture:

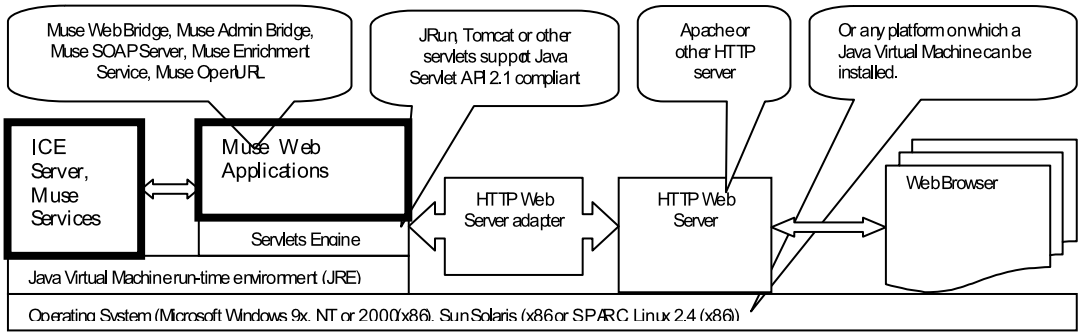


Muse can also run in an HTTP Web server environment if one is already installed or if there are scalability and loading issues that require its use.

HTTP Web server with servlets support compliant with Java Servlet API 2.1.

At the current writing the following HTTP servers have been tested with Muse:

Apache 1.3.19 or later and Apache 2.0.42 from Apache 2 series. See Muse External HTTP Server (Apache) for more details.



At the current writing the following servlets servers have been tested with Muse:

- Tomcat 3.2.x, 3.3.x, 4.x (starting 4.1.10), 6.0.x. See Muse External Servlets Engine (Tomcat) for more details.
- JRun 3.0.

Note that an HTTP server and a servlet server must work in combination, and not all HTTP server releases implement connectors for all servlets server releases. If you do not have an HTTP Web server with servlets support, you can obtain one from one of the following vendors:

Vendor	URL Address
Apache Tomcat	http://tomcat.apache.org/index.html
Apache	http://httpd.apache.org/
Allaire	http://www.adobe.com/products/jrun/

3.0

Installing Java Virtual Machine

This section describes the installation procedure for Java SE Development Kit (JDK).

The first step in this installation is to download the archive containing the Java SE Development Kit (JDK) for the specific platform to be used. The download address for the latest released version is `http://www.oracle.com/technetwork/java/javase/downloads/jdk6-downloads-1637591.html`.

Note: We currently recommend installing JDK 1.6 because it was throughly tested. Note that JDK version must be 1.6 at least Update 18, Muse does not work with JDK 1.5 or lower.

Note: You must obtain the Java Virtual Machine only from Oracle or IBM. Muse works if and only if the Java Virtual Machine is from Oracle or IBM. Note that there is no IBM JDK for Windows platform.

3.1 Windows Platform Installation

Starting from the web address indicated above download the appropriate `Windows (all languages, including English)` version - either for a 32 bit architecture or 64 bit. The downloaded file is a self-extracting archive presented as an executable file. After running the archive the only information the installation process will require is the directory location in which the JVM should be placed.

If you run into any problems during the installation, please check , depending on your architecture, the `Installation Instructions` section at:
`http://www.oracle.com/technetwork/java/javase/documentation/install-windows-152927.html` or
`http://www.oracle.com/technetwork/java/javase/documentation/install-windows-142126.html`.

Some Java programs require the definition of a supplementary environment variable called `JAVA_HOME` pointing to the JDK installation. If this is required, enter the following, assuming the destination directory

is `c:\jdk1.6.0` :

```
set JAVA_HOME=c:\jdk1.6.0
```

Setting the `JAVA_HOME` variable is mandatory on the Windows 2003 Server platform.

Note: For Apache Tomcat embedded within Muse to start on the Windows systems, the `JAVA_HOME` must be previously set, otherwise, the "`Neither the JAVA_HOME nor the JRE_HOME environment variable is defined\At least one of these environment variable is needed to run this program`" will be received.

3.2 UNIX Platform Installation

There are separate JDK Environments for Linux and Solaris. Each is accessible following the appropriate link from the web address indicated above: `Linux self-extracting file` or `Solaris SPARCTM/x86 self-extracting file.`

If you run into any problems during the installation, please check, depending on your architecture, the `Installation Instructions` section:

`http://www.oracle.com/technetwork/java/javase/install-linux-rpm-137089.html`

,

`http://www.oracle.com/technetwork/java/javase/install-linux-64-rpm-138254.html,`

`http://www.oracle.com/technetwork/java/javase/install-linux-self-extracting-138783.html or`

`http://www.oracle.com/technetwork/java/javase/install-linux-64-self-extracting-142068.html`

For Solaris, there are two versions of JDK available, according to the installation platform: SPARCTM or IntelTM x86 platform.

If you run into any problems during the installation, please check, depending on your architecture, the `Installation Instructions section:`

`http://www.oracle.com/technetwork/java/javase/install-solaris-139361.html`

or

`http://www.oracle.com/technetwork/java/javase/install-solaris-64-138849.htm`

l

If you install the JDK Environment into a system-wide location such as `/usr/local`, you must first become root to gain the necessary permissions. If you do not have root access, install the JDK Environment into your home directory or a subdirectory that you have permission to write to.

Some environment variables such as PATH and CLASSPATH (indicating the place where the Java files were installed) must be modified or set to use JDK efficiently.

Although this is not a strictly Java-related issue any longer, some programs require the definition of a sup plementary environment variable called `JAVA_HOME` pointing to the JDK installation. If this is required, enter the following, assuming the destination directory is `/usr/java` you should type:

```
export JAVA_HOME=/usr/java/jdk
```

Add the line above in the file `/etc/profile` to make this setting permanent on Linux or Solaris. The variable becomes globally available after the next login.

4.0

Installing Muse

Starting with Muse 2600 Muse Proxy will not be installed using Muse Setup. To install Muse Proxy server you have to run Muse Proxy Setup which needs a serial number. The serial number required for Muse Proxy Server is any serial number belonging to you, the only requirement being to have enabled the entry for Muse Proxy. Details about the install of the Muse Proxy Server and the install/start/uninstall/stop of the Muse Proxy Service are provided in the `${MUSE_HOME}/proxy/doc/Muse Proxy.pdf` manual.

4.1 Muse Installation Procedure

Although the `Muse Setup` installation package can be installed on virtually any platform that supports Java, there are a number of platform-specific issues that keep this platform-neutral installation package from providing a simple and reliable user experience.

Currently the setup installation package provides platform packs for:

Launcher Name	Platform Distributions
muse-linux-x86.bin	RedHat, Fedora, CentOS, SuSE, Mandriva, FreeBSD, Debian, Slackware, Gentoo
muse-solaris-sparc.bin	Sun Solaris Sparc v. 8-10
muse-solaris-x86.bin	Sun Solaris x86 v. 8-10
muse-win32.exe	Microsoft Windows 2000, Windows XP, Windows Server 2003, Windows 7, Windows Server 2008
muse-winIA64	64 bits Windows versions
muse-aix-power.bin	AIX on Power Architecture 4.3.2, 4.3.3, 5.0.0 & 5.1.0/5L
muse-macos.bin	Mac OS X 10.0-10.3, 10.4 PPC
muse-hpux11.bin	HP-UX 11.0, 11.i / 11.11
muse-hp1020.bin	HP-UX 10.20
muse-hpIA64.bin	64 bits HP-UX versions
muse-generic-unix.bin	Generic Unix; should be used for Unix platforms where no other

	specific install packages are available
muse-setup.jar	Java Virtual Machine 1.6 on the supported platforms

Muse can run on virtually any platform that supports Java. If the platform you need for your Muse in stallation is not listed above, please contact your Muse provider for further details.

The following issues should be noted if you are installing Muse as a normal user (without root/ administration rights):

The user installing Muse must have all permissions in the parent directory that are required to create the Muse installation directory and all the Muse files. By default Muse installation directory is `/opt/muse`. For example, if the user John is installing Muse under the `/opt/muse` directory, then John must have write and execute rights on the `/opt` directory. If rights to the parent directory cannot be given due to security reasons, change the Muse installation directory to a directory where rights can be assigned.

The lack of `root/administrator` rights may generate errors in the Muse Services Panel File. These can be ignored as the Muse Services can be installed later.

After installing Muse Products as a normal user (without `root/administration` rights), run Muse Services Setup as a `root/administrator` user.

The Muse installation is started in different ways, depending upon your operating system and hardware ar chitecture.

On Windows operating systems, double-click `setup.exe` to begin the setup graphical interface mode procedure. On UNIX platforms, run `setup.bin` file to begin the setup graphical interface mode procedure. (Check the name of the launcher file for the specific installation package and platform you are using.) See Section 4.1.1, “Launching using the Native Launcher” for more detailed references.

`Muse Setup` will automatically look for an acceptable Java Virtual Machine for the installation. The launcher will terminate if none of the recommended Java Virtual Machine can be found, so read the Chapter 3, *Installing Java Virtual Machine* before proceeding with the Muse Installation.

As an alternative to native platform launchers, a Java `.class` file you can execute from the command line is compressed in an appropriate `setup.jar` file. (Check the name of the archive file for the specific installation package you are using.) Use this installation method if none of the recommended Java Virtual Machines can be found and the launcher terminates.

This alternative installation can run in graphical mode using Swing (by default) or AWT, or in console mode.

The graphical user interface (GUI) consists in a series of panels that display to the user. To start the setup in graphical mode using the Swing interface use the following command line:

`java -cp setup.jar run` *(or alternatively* `java -jar setup.jar`*)*

To start the setup in graphical mode using the AWT interface use the following command line:

```
java -cp setup.jar run -awt
```

When the `Muse Setup` wizard is run in console mode, all messages and information are sent to the console instead of appearing on graphical panels. While running a wizard in console mode, you can move forward in the wizard by pressing Enter. To start the setup in console mode use the following command line:

```
java -cp setup.jar run -console
```

Command Line Options available for all types of installations (graphical, console, silent ,and from native executable):

-log !<log-file> @ALL Logs all the events in the specified log file. Note: on some Unix shells the character ! is interpreted by the shell and is not passed to the program. For those shells this character must be escaped by replacing it with \!.

-Dis.debug=1 Specifies to display all debug messages that occur at runtime on the Java console. For example:

```
java -Dis.debug=1 -cp setup.jar run
```

Once the `Muse Setup` installation is successfully completed the `Muse Applications Setup` in stallation package can be started, as described below, to add your own customized Application(s) if any have been supplied by your Muse provider.

When running with JDK 1.6 update smaller than 18 the `Muse Setup` may fail with `OutOfMemory` errors (these errors are written in the ${MUSE_HOME}/install.log file). If the upgrade of JDK to a version greater or equal with upgrade 18 is not possible then start the `Muse Setup` with higher memory values by adding into the command line to the `java` executable the following:

```
-Xmx512M -Xms256M
```

4.1.1 Launching using the Native Launcher

A native launcher wraps the Java archive file inside a platform-specific executable file.

Starting the native launcher acts the same as launching the command:

```
java -jar setup.jar -options-record <user_home>/muse-options.txt -options
<user_home>/muse-options.txt
```

Where `<user_home>` will be replaced with the path representing the home directory for the current user.

For successful installation an options file will be recorded and will preserve user's selections and input data. This file will be used and recreated for successful upgrades in the future.

Note: The options above require write access in the local directory where the options file is to be created.

Note: If an options file existed previously under `<user_home>/muse-options.txt`, it will be deleted when Muse Setup is started and will only be written upon a successful installation.

See the Section 4.1.4, "Running the Muse Setup Wizard in Silent Mode" for information about the above parameters.

Command Line Options Available for Native Launchers:

- `is:log <file name>` is useful for native launchers that hide the Java console. When this option is specified, all output that occurs in the Java console is logged to the specified file.
- `is:version` reports the version of the native launcher itself. When this option is specified, the launcher simply reports the version and exits without launching the application.
- `is:javaconsole` makes the Java console visible, overriding the value specified for the "Show Console" property when the launcher was built. Always use this option along with -console when you want to run the setup in console mode using the native launcher.
- `is:tempdir <directory>` allows the end user to specify a directory to which the launcher will write its temporary files. If the specified location either does not exist or is not a directory, the launcher will use the system temp directory instead, and will not give an error message. If this option is used for a launcher that executes a wizard (either an installer or an uninstaller), then the wizard will use the same directory for its temp files. The temporary directory used by the launcher must have all permissions for the user running the setup (rwx for Unix/Linux systems or Full Control on Windows systems). The temporary directory is usually located in "C:\Documents and Settings\<username>\Local Settings\Temp" for Windows systems and "/tmp" for Unix/Linux systems.
- `is:silent` prevents the display of the launcher UI to the user. This does not execute the wizard itself in silent mode.

- `is:javahome <Java home directory>` allows the end user to specifically tell the launcher the home directory location of the JVM. This option works only when executing a launcher for an installer or uninstaller wizard. In order for it to work, the JVM installed in the specified directory must be one of the JVMs specified in the "JVM Search Instructions" property of the launcher distribution.

4.1.2 Running the Muse Setup Wizard in Graphical Mode

To start the setup in graphical mode use the following command line:

`java -cp setup.jar run` *(or alternatively* `java -jar setup.jar`*)*

To advance in the installation procedure click the `Next` buttons. A `Back` button is available to review previous steps. The `Cancel` button, when available, cancels the installation process.

After the initial welcome panel, the next panel requires the product Serial Number. The Serial Number is not a random string of letter-digits conforming to an algorithm, but is an encrypted sequence that encodes data about the user's licensed Muse products. Each personalized Serial Number is established by the Muse provider and is delivered individually using different media (mail, e-mail, floppy disk, etc.). The same installation package might be used to install all Muse products according to the Serial Number, depending upon the products licensed from the Muse provider.

Take care to enter the serial number exactly as supplied. They are case sensitive. Be careful to distinguish between 'one', 'el', 'eye', 'zero', and 'oh' as any of these may occur in a Serial Number.

A license agreement follows the product Serial Number panel. The user accepts the license terms by checking the appropriate text, `I accept the terms of the license agreement` (acceptance is not assumed by default).

The next panel gives a brief description of Muse functionality.

A Muse installation can be an initial install or an installation over an existing Muse version as part of a repair or an upgrade procedure. In a first-time installation, a new directory structure must be defined, and the next step of the setup process when this is the case establishes the destination directory of Muse in the file system. By default the installation process will set up and install the package under `C:/Program Files/muse` directory for Windows and under `/opt/muse` directory for Solaris and Linux. This path can be changed during an initial installation process.

After the destination directory name is set, the next step asks the user to choose between a `Custom` and a `Typical` installation. A `Custom` installation allows the user to select the components to install, as

applicable for their licensed Muse products. A `Typical` installation is recommended.

Currently, `Muse Setup` package includes the following components:

- `Information Connection Engine (ICE) Server`: this is the core system, providing a framework for data Retrieving and Processing Modules.
- `Information Connection Engine Tools`: utility tools for maintenance, configuration, and administration of the ICE Server.
- `Muse Web Bridge`: Web base client, actually a set of servlets and associated HTML pages.
- `Muse Admin Bridge`: The core admin API required for Muse Console web applications to work. These are administrative tools used by administrators of Muse environments.
- `Muse Default Applications`: default Applications for running the Muse search interface, which allow a quick start immediately after installation. Other Muse Applications can be added or created at a later time. A different Muse Applications Setup package is used to add or update existing customized Application(s) if any have been supplied by your Muse provider.
- `Muse Z39.50 Bridge`: this is a Z39.50 server, which allows applicable Muse data sources to be searched using this standard protocol.
- `Muse SOAP Bridge`: this is a SOAP server, which allows applicable Muse data sources to be searched using this standard protocol. SOAP messages are transmitted over HTTP protocol. In addition to the standalone server, a SOAP Servlet is also installed.
- `Apache Tomcat embedded within Muse`: this is the servlet container that is used in the official Reference Implementation for the Java Servlet and JavaServer Pages technologies. The Java Servlet and JavaServer Pages specifications are developed by Sun under the Java Community Process.
- `Muse HTTP Server`: this is an HTTP server, supporting the HTTP 1.0 version protocol.

  Note: Muse HTTP Server is now deprecated and Apache Tomcat embedded within Muse must be used instead.
- `Muse Enrichment Service`: Muse Enrichment Service is a server-side Web application aimed for ISBN queries but other query attributes can be added for later extensions. Given an ISBN, it outputs enriched information about the item having that ISBN (or an error message in cases of wrong usage or configuration). It works against a local set of indexes generated out of different databases.
- `Muse OpenURL`: This includes a Web OpenURL Generator that provides a mechanism for encoding a citation for an information resource, typically a bibliographic resource, as a URL.
- `Muse Serial Number Encoder`: This is an internal MuseGlobal utility used to generate appropriate Serial Numbers. This component is not installed outside of MuseGlobal offices or without the express permission of MuseGlobal.
- `Muse Source Factory and Consoles`: These MuseGlobal services are comprised of a number of components. These tools, which run through a browser interface, permit Muse users, partners, and partner's users to easily search the MuseGlobal resource library, configure Ap plications, download and install Sources into Applications, apply fixes for failed Sources, obtain en

hancements for working Sources, and perform many other functions.

- **Muse Control Center**: A utility tool used by the administrator of Muse to test various components of the Muse environment or perform certain actions based on scheduled tasks.
- **Muse Builder**: A tool that automates many of the redundant activities of Muse Application developers. Automating these activities leads to better management of the Muse Applications, less development time, and better support for clients.
- **Muse XML DB Management System**: Provides a method of storing XML documents in a compatible and standardized format through the XMLDB interface http://www.xmldb.org .The Muse Personal Profiles Management System, used for managing patron personal data, is such an interface.
- **Muse Statistics Monitor**: This tool produces various statistics, charts, and reports based on the statistics logs produced by the ICE Server. Examples of statistics analyses are: Memory Usage, Session Count, Network Analysis, and Connector Activity.
- **Muse Web Bridge Communication Interface Kit**: includes testXMLAPI tool and Muse Web Bridge Communication Interface.pdf manual.
- **Muse Admin Bridge Communication Interface Kit:** include MuseAmin Client tool (this is a tool that connects to a Muse Admin via TCP/IP and executes MuseAdmin commands listed in an XML document received as input) and MuseAdmin Communication Interface.pdf manual.

Customer-specific bridges that allow intersystem communication between third party systems and Muse.

The **Typical** installation selects all licensed components by default.

If this is an upgrade and not a first-time installation, by clicking **Next** the setup program detects which Muse Services are associated with the selected products already installed on the target machine. If no Muse Service is found the setup automatically moves to the next panel. It is recommended that the user first uninstall all installed Services by unchecking the checkboxes associated with them in the panel. Otherwise some files used by these services will not be updated when Muse Setup re-installs them.

Clicking **Next** the setup program displays a status panel showing the location, the components selected, and the amount of space that will be required.

A final **Next** starts the installation process.

Note: If the user does not have administrator rights and Muse Services are already installed (an upgrade is being performed), a panel with instructions on how to stop/uninstall Muse Services will display and the setup will end.

Once all components have been copied, the user can choose which Muse Services should start automatically during bootup. This option is available only for Muse Services selected for installation. You must have administrator rights to perform this operation. If you do not have administrator rights, a panel with instructions on how to install/uninstall/start/stop Muse Services will display, but you need root/

administrator rights to carry out these instructions - see Section 4.1.6, “Running the Muse Services Setup”.

If you do not select any Muse Service for automatic startup, you can still manually start Muse Services or use Muse Services Setup to set this up at a later time.

If the `Muse Console for Applications Administration` has been installed, then appropriate passwords must be set. These passwords are used to access the account for system administration and maintenance. An IP or an address pattern from which the administrator can access the `Muse Console for Applications Administration` can also be set at this time.

At this point the `Product Registration Form` must be filled in. This step requires direct Internet access to deliver the completed form online to MuseGlobal. All information provided in the form is for MuseGlobal internal use only, and will not be disclosed to any third party. Uncheck the box on the form if you don’t want to receive any emails from MuseGlobal, and your email address will not be added to our mailing list. This form must be successfully delivered to complete the installation. To run the re gistration form at a later time, the `-goto registration` parameters can be used in the setup package command line (e.g. `java -jar setup.jar -goto registration).`

If online registration fails for any reason or you don’t want Muse Setup to send it to MuseGlobal online, save the `Product Registration Form` for later usage and send it by email, mail, facsimile, or whatever form is most convenient, to:

```
MuseGlobal, Inc. 222 Kearny Street Suite 550 San Francisco, CA 94108 U.S.A.
Email: support@museglobal.com
Tel: + 1 415 896 MUSE
Fax: + 1 415 896 6870
```

Successful online registration allows Muse to be started upon successful completion of the installation. Offline registration requires a new signature and additional instructions provided by MuseGlobal to start Muse.

Re-login is required on UNIX platforms.

4.1.3 Running the Muse Setup Wizard in Console Mode

To start the setup in console mode use the following command line:

```
java -cp setup.jar run -console
```

The console mode translates into text the data presented in graphical panels. While running the wizard in

console mode, you can move forward in the wizard by pressing `Enter` key.

Follow the onscreen instructions and read the equivalent Graphical Mode instructions.

4.1.4 Running the Muse Setup Wizard in Silent Mode

To start the setup in silent mode, use the following command line:

```
java -jar setup.jar -options <options file> -silent
```

The silent mode install does not interact with the user unless an error occurs. If an error occurs, a message displays and the setup exits.

The silent mode install uses a pre-built `<options file>,` which contains the options needed by the wizard to run, just like the user would enter them.

Note: this mode will override the built-in path (`<user_home>/muse-options.txt`) *of the options file with the path that you supply.*

To start the setup, use the following command line to generate the options file:

```
java -jar setup.jar -options-record <options file>
```

Note: This will override the built-in `path (<user_home>/muse-options.txt)` *of the options file with the path that you supply at this time.*

This runs a normal setup where all the installation options are recorded into the `<options file>` as specified.

There is an additional parameter regarding the creation and use of options files.

```
java -jar setup.jar -options-template <options file>
```

All the properties that can be specified to the wizard will be written in the specified file. This file is a template file and the user must set the proper values for the properties. Later this file can be used as an input options file.

Note: All the options above require write access in the directory where the options file is to be created.

4.1.5 Running the Muse Application Setup

This installation procedure is quite simple and is similar to the `Muse Setup` installation. Installation also provides platform packs suffixed as above.

This setup program installs Muse Applications on your computer only if `Muse` has been already installed. If an installed copy of Muse is not found, either because it was originally installed manually (not using `Muse Setup`) or the InstallShield registry which stores information about Muse software products has been accidentally removed, it still gives you the chance to manually select the installation directory of Muse if one is installed.

By default the installation process will set up and install the package under `C:/Program Files/muse` directory for Windows and under `/opt/muse` directory for Solaris and Linux.

This setup program also allows you to choose between the different Muse Applications available with the package according to the supplied Serial Number, as well as to choose what components to install for the selected Muse Applications. (For instance, uncheck the `Web Interface` component to preserve existing HTML pages displayed in the browser by the `Muse Web Bridge` if you have changed them from the default versions.)

4.1.6 Running the Muse Services Setup

If `Muse Services` haven't been selected during `Muse Setup` installation, you can still set up which services are to be started automatically when the system boots up by running the Muse Services Setup. Muse Services Setup is also used to remove services from automatic startup. Administrator rights are required to manage Muse Services.

Under a graphical environment Muse Services Setup can be started from Start menu (`Start/Programs/Muse/Muse Services Setup`).

`Muse Services Setup` can also be started by running `C:\Program Files\muse\setup\startMuseServicesSetup.bat` on a Windows machine, assuming the Muse is installed under the `C:\Program Files\muse` directory, or by running `/opt/muse/setup/startMuseServicesSetup` on a Unix machine, assuming the Muse is installed under the `/opt/muse` directory.

If you want to perform Muse Services Setup in console mode, run the files mentioned above with the `-console` parameter. E.g.:

```
C:\Program Files\muse\setup\startMuseServicesSetup.bat –console
```

When `Muse Services Setup` is started a panel displays, listing the Muse Services and their install status. Check (or uncheck) the checkbox near each service name to indicate whether that service should (or should not) start automatically when the system boots up. Click the Finish button to save the new settings.

Note: The user must have administrator rights to set up the services. If the Muse Setup determines this is not the case, the user will receive information on how to install Muse services. The user will need administrator rights to successfully complete Muse Services Setup.

The following steps should be performed to set up Muse Services after Muse has been installed as a normal user:

1 On Unix Systems

Log in as root

In the root home directory (you should there after root login), create a symbolic link for the Install Shield directory. This is located under the home of the user used for initial Muse Installation. To do this, launch the following command, substituting ${USER_HOME} with the full path to the home of the user used for initial Muse Installation:

```
ln -s ${USER_HOME}/InstallShield InstallShield
```

1 Launch the following shell script, substituting ${MUSE_HOME} with the full path to the Muse home directory (e.g. /home/john/muse):

```
${MUSE_HOME}/setup/startMuseServicesSetup.sh
```

1 On Windows Systems

Log in as administrator (or as an user having administrator rights)

Give administrator rights to the user used for the initial Muse Installation (For example on Windows XP: `Control Panel -> User Accounts -> <<User Name>> -> Change the account type -> Select computer administrator.`

Logoff

Log in as the user used for the initial Muse Installation.

Go to the setup subdirectory in Muse installation directory and launch the Muse Services Setup - `startMuseServicesSetup.bat`

Remove the administrator rights for this user and logoff.

4.1.7 Running the Muse Registration Setup

If Muse Registration was not performed during Muse Setup installation, you can still register with

MuseGlobal, Inc.

In a graphical environment Muse Registration Setup can be started from Start menu (`Start/Programs/Muse/Muse Registration Setup`).

`Muse Registration Setup` can also be started by running on a Windows machine `C:\Program Files\muse\setup\startMuseRegistrationSetup.bat` assuming the Muse is installed under the `C:\Program Files\muse` directory, or by running on a Unix machine `/opt/muse/setup/startMuseRegistrationSetup.sh, /opt/muse/setup/startMuseRegistrationSetup.csh, depending on the version of your shell,` assuming the Muse is installed under the `/opt/muse` directory.

4.2 Product Registration

The installation must be registered before the `Information Connection Engine Server` can be started. This can be done during installation or at a later time.

The `Product Registration Form` must to be filled in to register the installation. This step requires direct Internet access to deliver the completed form online to MuseGlobal. Information provided in the form is for MuseGlobal internal use only, and will not be disclosed to any third party. Uncheck the box on the form if you don't want to receive any emails from MuseGlobal, and your email address will not be added to our mailing list. This form must be successfully delivered to complete the installation.

To run the registration form at a later time, use `the Muse Registration Setup` (details for starting it can be found in the `Running Muse Registration Setup` section of this manual) or enter the `-goto registration` parameters in the setup package command line (e.g. `java -jar setup.jar -goto registration`).

If online registration fails for any reason or you don't want Muse Setup to send it to MuseGlobal online, save the `Product Registration Form` for later usage and send it by email, mail, facsimile, or whatever form is most convenient, to:

```
MuseGlobal, Inc. 222 Kearny Street Suite 550 San Francisco, CA 94108 U.S.A.
Email: support@museglobal.com
Tel: + 1 415 896 MUSE
Fax: + 1 415 896 6870
```

Successful online registration allows Muse to be started upon successful completion of the installation. Offline registration requires a new signature and additional instructions provided by MuseGlobal to start

Muse.

4.3 Rerunning the Muse Setup

You can run the `Muse Setup` again to install any components you did not previously install on your system or to repair or uninstall an installed component.

Rerunning Muse Setup is also required if a new Serial Number is provided for additional licensed Muse products.

4.4 Muse Upgrade Procedure

When you run a different Muse Setup than the one originally installed on your machine you will go through an upgrade procedure. This is similar to the first-time installation, but when Muse is installed in the same location more than once product components can determine whether or not to replace existing files.

Note: *It is highly recommended that all Muse Services be stopped during an upgrade procedure. The setup will not continue if an upgrade is performed on an account without administrator privileges and installed Muse Services are detected.*

During the upgrade you may be prompted to replace or keep some files that were changed during the life of the old installation. You also have the option to replace all without being prompted.

Normally, almost all of the files will be overwritten if the components they belong to have been selected for install. Only those files that contain custom information about users and security issues will be left unchanged.

Once the upgrade procedure has been completed, continue with the instructions in `${MUSE_HOME}/Upgrades.txt` that lists required manual changes from an old to a newer release version.

4.4.1 Muse Software Upgrade

The Muse software packages will overwrite your existing Muse software, unless it finds a newer file, in which case you will be prompted for the action you wish to take. Choosing a `Custom` installation allows for selective upgrading, but unless instructed otherwise, you will normally be installing everything

provided in your software installation package. Muse configuration files are left untouched, so the settings for any Applications or Users you have added will be safeguarded.

Additional steps may be required when upgrading between certain release versions, but these instructions will be provided separately.

4.4.2 Application Upgrade

As above, Muse Application upgrade packages will overwrite existing Applications with newer files. Unless you want to completely overwrite all Applications with the upgrades supplied you should select the `Custom` installation option. This allows for the optional upgrading of individual Applications and individual components within the Application. In either case it is always a sensible precaution to back up all files in the Applications home directory before starting an upgrade.

Additional steps may be required when upgrading between certain release versions, but these instructions will be provided separately.

4.5 Uninstalling Muse

To uninstall the product, go to the command line and navigate to the directory in which Muse was installed. Under Windows, run `uninstall.bat` and for Solaris or Linux run `uninstall.sh` or `uninstall.csh` to begin the graphical interface mode uninstall procedure. If you want the uninstall to occur in console mode, run the above files with the `-console` parameter. E.g.:

```
Uninstall.bat -console
```

On other systems, run the `uninstall.jar` file, which can be found in the uninstall subfolder. Use the following command line to begin the graphical interface mode procedure:

`java -cp uninstall.jar run` *(or alternatively* `java -jar uninstall.jar`*)*

Use the following command line to begin the console interface mode procedure:

```
java -cp uninstall.jar run -console
```

On a Windows system, you can select `Uninstall Muse` from the Muse folder on your `Start/Programs` menu.

Windows 7 UAC (User Account Control) strips away any `"administrative"` power from ap plications, tasks, features, or actions that a user performs during routine functionality. There are 4 configurable permissions levels in UAC, the default one being `"Notify me only when programs`

`try to make changes to my computer"`. When using the default UAC configuration, the uninstall of Muse will give may errors of the form `"C:\Program Files\muse\wizard.log (Access is denied)"`, assuming `C:\Program Files` is the location where Muse is installed. If you are experiencing such problems, set the `UAC` level to `"Never Notify"` before uninstall Muse. After Muse is uninstalled you can restore the UAC default setting or whatever needed.

During uninstall you may be prompted to remove or keep some files that were changed during the life of the installation. You also have the option to remove them all without being prompted.

5.0

Starting Muse Services

Various Muse services can be automatically or manually started, depending on the functionality your particular installation requires or your license with your Muse provider.

For customer-specific bridges that allow intersystem communication between third party systems and Muse, consult the appropriate section in the documents provided with that specific product.

It is possible to start/stop each service individually or start/stop all services at once.

For each such operation and for each service, platform- and shell-specific scripts are provided: one for the Windows command prompt (file name ending in `.bat`), one for C-Shell (file name ending in `.csh`) and one for Bash (no file extension). The first is for Windows environments, so they will not be present on Unix installations, while the last two are for UNIX environments and will not be present on Windows installations.

After starting any of the Muse Services from a remote location (for example, using a `ssh` or a `telnet` terminal) make sure you properly exit the Console by calling the `exit` command. If the Console is not exited correctly or it times out the services started from that Console might terminate without notice.

Note: *Before starting any of the Muse services, make sure that the Muse environment variables are defined, and, if not, define them. The following are the Muse variables along with their values for a default installation:*

`MUSE_HOME` – This is the Muse installation directory. By default, it takes the following values, depending on the Operating System:

- On Windows `%MUSE_HOME%` is by default `C:\Program Files\muse`
- On Unix/Linux $MUSE_HOME is by default /opt/muse

`USE_HOME` – The Universal Search Engine (USE) installation directory. USE is the fundamental application within Muse. It provides all the functionality for each of the users. Default installation locations:

- On Windows `%USE_HOME%` is by default `C:\Program Files\muse\use`

- On Unix/Linux $USE_HOME is by default /opt/muse/use

`ICE_HOME` - This is the Information Connection Engine installation directory. By default it is installed in a directory located under:

- On Windows `%ICE_HOME%` is by default `C:\Program Files\muse\use\ice`
- On Unix/Linux $ICE_HOME is by default /opt/muse/use/ice

`MODULES_HOME` - This directory contains Core Modules and connector related files. Default installation locations:

- On Windows `%MODULES_HOME%` is by default `C:\Program Files\muse\use\modules`
- On Unix/Linux `$MODULES_HOME` is by default `/opt/muse/use/modules`

If the above variables are not defined they must be defined according to the Operating System Environment:

- On Windows this is usually done under `Control Panel -> System -> Advanced -> Environment Variables`

- On Unix this is done in the user profile in its home directory (if Muse was installed as a normal user) and/or in the global profile (e.g. `/etc/profile`) if Muse was installed as a root user.

The following sections describe how to start all the Muse Services at once (either at boot time or later) and how to start/stop individual Muse Services.

5.1 Starting all Muse Services at Boot Time

If you want Muse Services to start/stop at once or to automatically start all of the Muse Services at boot time you may have already selected `Install Muse Services` option from the installation or `Running the Muse Services Setup.`

If not, make the following changes.

On Linux, if you would like Muse Services to start at once or automatically when the machine starts, do the following:

```
ln -s /etc/init.d/muse /etc/rc.d/rc3.d/S99Muse
```

On Solaris, if you would like Muse Services to start at once automatically when the machine starts, do the

following:

```
ln -s /etc/init.d/muse /etc/rc3.d/S99Muse
```

The `rc3.d` directory used above supposes that your current run level is 3 (multi-user mode). If you want different run level, use the appropriate directory. For example, if you want run level 4 (X11 with KDM/GDM/XDM) use the directory `rc4.d`.

The above command creates a link to `muse` script that starts Muse services. This will cause Muse services to run at system start under the root user.

The `muse` script should be created by the installation program under `/etc/init.d` directory. If the in stallation is done manually one should be available under `${ICE_HOME}/muse`. Be sure to update the required variables inside the `muse` script (e.g. the `MUSE_HOME` variable).

The `muse` script can also be used to start/stop all Muse services simultaneously at any time after the Operating System is booted:

- Use `/etc/init.d/muse` or `/etc/init.d/muse start` to start the all Muse Services at once

- Use `/etc/init.d/muse stop` to stop all Muse Services at once

- Use `/etc/init.d/muse restart` to restart all Muse Services at once

On Windows NT, 2000, XP if you would like Muse Services to start up automatically when the machine starts, check the individual instructions above regarding Windows Services in the paragraph Running the Muse Services Setup.

Note: *Muse Services started at boot time on Unix/Linux systems using the above methods will start and run as the user owning the Muse install directory (${MUSE_HOME}).*

You may use this script to start all the Muse services during boot time or anytime during your login session.

If you want to start Muse as a different user than the one owning the Muse installation directory, edit `/etc/init.d/muse` and comment the line that automatically detects this user. Then specify the desired user on the line below the self explanatory comment:

```
# who is the owner of the Muse install directory?
MUSE_OWNER=`ls -dl "${MID}" | awk '{ print $3 }'`
```

If you want to start Muse services as another user than the owner of the installation directory, comment the line above (the one that instantiates the MUSE_OWNER variable), and add a line like the one below and specify the desired user:

```
MID_OWNER=specify_a_user
```

You may also use this script to start all the Muse services at once as an user other than the one you used when you logged in. To do that, run the script:

```
/etc/init.d/muse
```

If you are not root and Muse is installed as an user other than root, the script will prompt you to enter the password for the user to be used when the script is run.

5.2 Starting/Stopping the Information Connection Engine (ICE) Server

Once the installation is complete and Java is installed on your machine (see Chapter 3, *Installing Java Virtual Machine*), check the ICE server installation.

When running inside the Microsoft Windows environment, you can start the ICE server manually using `startServer.bat` from `c:\Program Files\muse\use\ice` assuming the default installation directory.

To stop the ICE server use `stopServer.bat` from `c:\Program Files\muse\use\ice` assuming the default installation directory.

On UNIX, if the default directory is used for installation (`/opt/muse`), use the following Bash Shell command script to start the ICE server from the command line:

```
/opt/muse/use/ice/startServer
```

To stop the ICE Server from the command line, use the following command:

```
/opt/muse/use/ice/stopServer
```

C-Shell scripts are also available for UNIX platforms.

Under a graphical environment the ICE server can be started from the Start menu `(Start/Programs/Muse/Information Connection Engine/Start Up ICE Server)` and can be stopped from the same Start menu `(Start/Programs/Muse/Information Connection Engine/Shut Down ICE Server).`

For Windows NT, 2000, XP, 2008 and 7, the ICE Server is also available as a Windows service. To use it this in this mode, control it by using only Windows Services management functions such as the `net`

command or the Services applet from the Windows Control Panel.

To start the ICE Server use the following command:

```
net start "Muse ICE Server"
```

Note that this Windows Service is set to the `Automatic` startup type. This means that it will automatically start during Windows initialization, before anyone logs on.

To stop the ICE Server from the command line use the following command:

```
net stop "Muse ICE Server"
```

You can also use the Services applet from the Windows Control Panel to start or stop the ICE Server.

Note: *It is very important that once it is started as an Windows Service the ICE Server should be stopped only by the Windows methods above (i.e. do not use the* `stopServer.bat` *control scripts or the Start menu items).*

The standard output and standard errors for the ICE Server run as a Windows Service are redirected to `${ICE_HOME}/ICEService.out` and `${ICE_HOME}/ICEService.err`, respectively.

5.3 Starting/Stopping the Muse Z39.50 Bridge

Once the installation is complete and Java is installed on your machine (see Chapter 3, *Installing Java Virtual Machine*), check the Muse Z39.50 Bridge installation.

When running inside the Microsoft Windows environment, you can start Muse Z39.50 Bridge manually using `startZBridge.bat` from `c:\Program Files\muse\z3950` assuming the default installation directory.

To stop Muse Z39.50 Bridge use `stopZBridge.bat` from `c:\Program Files\muse\z3950` assuming the default installation directory.

On UNIX, if the default directory is used for installation (`/opt/muse`), use the following Bash Shell command script to start Muse Z39.50 Bridge from the command line:

```
/opt/muse/z3950/startZBridge
```

To stop Muse Z39.50 Bridge from the command line, use the following command:

```
/opt/muse/z3950/stopZBridge
```

C-Shell scripts are also available for UNIX platforms.

Under a graphical environment Muse Z39.50 Bridge can be started from the Start menu `(Start/Programs/Muse/Z39.50 Bridge/Start Up Z39.50 Bridge)` and can be stopped from the same Start menu `(Start/Programs/Muse/Z39.50 Bridge/Shut Down Z39.50 Bridge).`

For Windows NT, 2000, and XP the Muse Z39.50 Bridge is also available as a Windows service. To use it in this mode, control it by using only Windows Services management functions such as the `net` command or the Services applet from the Windows Control Panel.

To start the Muse Z39.50 Bridge use the following command:

```
net start "Muse Z39.50 Bridge"
```

Note that this Windows Service is set to the `Automatic` startup type. This means that it will automatically start during Windows initialization, before anyone logs on.

To stop the Muse Z39.50 Bridge from the command line use the following command:

```
net stop "Muse Z39.50 Bridge"
```

You can also use the Services applet from the Windows Control Panel to start or stop the Muse Z39.50 Bridge.

Note: It is very important that once it is started as an Windows service the Muse Z39.50 Bridge should be stopped only by the Windows methods above (i.e. do not use the `stopZBridge.bat` *control scripts or the Start menu items).*

The standard output and standard errors for the Muse Z39.50 Bridge run as a Windows Service are redirected to `${MUSE_HOME}/z3950/MuseZ3950Service.out` and `${MUSE_HOME}/z3950/MuseZ3950Service.err`, respectively.

5.4 Starting/Stopping Apache Tomcat embedded within Muse

Once the installation is complete and Java is installed on your machine (see Chapter 3, *Installing Java Virtual Machine*), check the Apache Tomcat Server installation.

If you choose to use Apace Tomcat embedded within Muse as your servlets engine, then you will need the Java Virtual Machine Software Development Kit(JDK).

When running in the Microsoft Windows environment, you can start the Apache Tomcat embedded

within Muse manually by using `startup.bat` from `c:\Program Files\muse\tomcat\bin` assuming the default installation directory.

For Apache Tomcat embedded within Muse to start on the Windows systems, the `JAVA_HOME` must be previously set, otherwise, the `"Neither the JAVA_HOME nor the JRE_HOME environment variable is defined\At least one of these environment variable is needed to run this program"` will be received.

To stop Apache Tomcat embedded within Muse use `shutdown.bat` from `c:\Program Files\muse\tomcat\bin` assuming the default installation directory.

On UNIX, if the default directory is used for installation (`/opt/muse`), use the following Bash Shell command script to start Apache Tomcat embedded within Muse from the command line:

```
/opt/muse/tomcat/bin/startup.sh
```

To stop Apache Tomcat embedded within Muse from the command line, use the following command:

```
/opt/muse/tomcat/bin/shutdown.sh
```

To operate Muse Web Applications, the Apache Tomcat embedded within Muse starts by default with Security Manager.

Under a graphical environment Apache Tomcat embedded within Muse can be started from the Start menu `(Start/Programs/Muse/Apache Tomcat/Start Apache Tomcat)` and can be stopped from the same Start menu `(Start/Programs/Muse/Apache Tomcat/Shut Down Apache Tomcat).`

For Windows NT, 2000, and XP the Apache Tomcat embedded within Muse is also available as a Windows service. To use it in this mode, control it by using only Windows Services management functions such as the `net` command or the Services applet from the Windows Control Panel.

To start the Apache Tomcat embedded within Muse use the following command:

```
net start "EmbeddedApacheTomcat"
```

Note: This Windows Service is set to the `Automatic` startup type. This means that it will automatically start during Windows initialization, before anyone logs on.

To stop the Apache Tomcat embedded within Muse from the command line use the following command:

```
net stop "EmbeddedApacheTomcat"
```

You can also use the Services applet from the Windows Control Panel to start or stop the Apache Tomcat embedded within Muse.

Note: It is very important that once it is started as a service the Apache Tomcat embedded within Muse should be stopped only by the above Windows methods.

On a successful installation, entering the address `http://localhost:8000/` in a Web browser should display the Apache Tomcat Server embedded within Muse welcome page. (Port 8000 is the default listening port for Apache Tomcat embedded within Muse.)

5.5 Starting/stopping Muse HTTP Server

Note: Muse HTTP Server is now deprecated and Apache Tomcat embedded within Muse must be used instead.

Once the installation is complete and Java is installed on your machine (see Chapter 3, *Installing Java Virtual Machine*), check the Muse HTTP Server installation.

When running in the Microsoft Windows environment, you can start the Muse HTTP Server manually by using `startHTTPServer.bat` from `c:\Program Files\muse\http` assuming the default in stallation directory.

To stop Muse HTTP Server use `stopHTTPServer.bat` from `c:\Program Files\muse\http` assuming the default installation directory.

On UNIX, if the default directory is used for installation (`/opt/muse`), use the following Bash Shell command script to start Muse HTTP Server from the command line:

```
/opt/muse/http/startHTTPServer
```

To stop Muse HTTP Server from the command line, use the following command:

```
/opt/muse/http/stopHTTPServer
```

C-Shell scripts are also available for UNIX platforms.

To operate Muse Web Applications, the Muse HTTP Server it has to be started with Security Manager using `-security` parameter in the command line as above.

Under a graphical environment Muse HTTP Server can be started from the Start menu `(Start/Programs/Muse/HTTP Server/Start Up HTTP Server)` and can be stopped from the

same Start menu `(Start/Programs/Muse/HTTP Server/Shut Down HTTP Server).`

For Windows NT, 2000, XP, 2008 and 7 the Muse HTTP Server is also available as a Windows service. To use it in this mode, control it by using only Windows Services management functions such as the `net` command or the Services applet from the Windows Control Panel.

To start the Muse HTTP Server use the following command:

```
net start "Muse HTTP Server"
```

Note that this Windows Service is set to the `Automatic` startup type. This means that it will automatically start during Windows initialization, before anyone logs on.

To stop the Muse HTTP Server from the command line use the following command:

```
net stop "Muse HTTP Server"
```

You can also use the Services applet from the Windows Control Panel to start or stop the Muse HTTP Server.

Note: *It is very important that once it is started as a service the Muse HTTP Server should be stopped only by the above Windows methods (i.e. do not use the* `stopHTTPServer.bat` *control script or the Start menu items).*

The standard output and standard errors for the Muse `HTTP Server` run as a Windows Service are redirected to `${MUSE_HOME}/http/MuseHTTPService.out` and `${MUSE_HOME}/http/MuseHTTPService.err`, respectively.

On a successful installation, entering the address `http://localhost:8000/` in a Web browser should display the Muse HTTP Server welcome page. (Port 8000 is the default listening port for Muse HTTP Server.)

6.0

Starting Muse Web Applications

A Web application is a dynamic extension of a Web server. Web components are supported by the services of a runtime platform called a Web container or Servlets Engine (Apache Tomcat embedded within Muse, External Apache, JBoss, GlassFish etc.). This chapter covers the installation or deployment of the Muse Web Application into a Web container and the access to a URL that references that Web Application.

Certain aspects of Web application behavior can be configured when the Application is deployed. The configuration information is maintained in a text file in XML format (`web.xml`) called a Web application deployment descriptor, which is pre-configured during the main installation process.

The below desktop web browsers were used to test the Muse Web Applications for the latest Muse version:

- Internet Explorer 10.0,
- Mozilla Firefox 27.0.1,
- Google Chrome 33.0;

6.1 Starting the Muse Web Bridge

The Muse Web Bridge is a session-oriented Web application. Once the installation is complete and the HTTP server with servlets support has been installed and configured on your machine, check the Muse Web Bridge installation.

The Muse Web Bridge can be run under any HTTP Web server(e.g.: Apache Tomcat embedded within Muse) with servlets support.

The Muse Web Bridge is already configured with a specific context for this application when it runs under the Apache Tomcat embedded within Muse.

On a successful installation, entering the address `http://localhost:8000/muse/` in a Web browser should display the Muse Web Bridge page.

The Muse Web Bridge can be used from any Web Browser, its requirements being determined by the different Muse Applications. In a graphical environment the Muse Web Bridge can be used from the Start menu `(Start/Programs/Muse/Muse Web Bridge).`

6.2 Starting the Muse Admin Bridge

The Muse Admin Bridge is a session-oriented Web application. Once the installation is complete and Java is installed on your machine (see Chapter 3, *Installing Java Virtual Machine*), check the Muse Admin Bridge installation.

Any Muse Admin Bridge can be run under any HTTP Web server(e.g.: Apache Tomcat embedded within Muse) with servlets support.

The Muse Admin Bridge is already configured with a specific context for this application when it runs under Apache Tomcat embedded within Muse.

On a successful installation, entering the address `http://localhost:8000/mmc/` in a Web browser should display the Muse Admin Bridge page.

In a graphical environment applicable Muse Console(s) can be used from the Start menu `(Start/Programs/Muse/Muse Admin Bridge).`

See the Muse External HTTP Server and Muse External Servlets Engine manuals, depending on the engine required, if another HTTP Web server must be used.

6.3 Starting the Muse SOAP Bridge

The Muse SOAP Bridge is a session-oriented Web application. Once the installation is complete and Java is installed on your machine (see Chapter 3, *Installing Java Virtual Machine*), check the Muse SOAP Bridge installation.

Muse SOAP Bridge can be run under any HTTP Web server(e.g.: Apache Tomcat embedded within Muse) with servlets support.

The Muse SOAP Bridge already comes configured with a specific context for this application when it

runs under the Apache Tomcat embedded within Muse.

On a successful installation, entering the address `http://localhost:8000/soap/` in a Web browser should display the Muse SOAP page.

In a graphical environment the Muse SOAP Bridge can be tested from the Start menu `(Start/Programs/Muse/Muse SOAP Bridge/SOAP Bridge Start Page).`

See the Muse External HTTP Server and Muse External Servlets Engine manuals, depending on the engine required, if another HTTP Web server must be used.

6.4 Starting the Muse Enrichment Service

This is a session-oriented Web application. Once installation is complete and Java is installed on your machine (see Chapter 3, *Installing Java Virtual Machine*), check the Muse Enrichment Server installation.

The Muse Enrichment Server can be run under any HTTP Web server(e.g.: Apache Tomcat embedded within Muse) with servlets support.

The Muse Enrichment Server already comes configured with a specific context for this application when it runs under the Apache Tomcat embedded within Muse.

On a successful installation, entering the address `http://localhost:8000/enrich/` in a Web browser should display the Muse Enrich page.

In a graphical environment the Muse Enrichment Service can be tested from the Start menu `(Start/Programs/Muse/Muse Enrichment Service/Muse Enrichment Service).`

See the Muse External HTTP Server and Muse External Servlets Engine manuals, depending on the engine required, if another HTTP Web server must be used.

6.5 Starting the Muse OpenURL

This is a session-oriented Web application. Once installation is completed and Java is installed on your machine (see Chapter 3, *Installing Java Virtual Machine*), check the Muse OpenURL installation.

The Muse OpenURL Generator can be run under any HTTP Web server(e.g.: Apache Tomcat embedded within Muse) with servlets support.

The Muse OpenURL Generator already comes configured with a specific context for this application when it runs under the Apache Tomcat embedded within Muse.

On a successful installation, entering the address `http://localhost:8000/openurl/` or `http://localhost:8000/openurl`/generator in a Web browser should display the Muse OpenURL Generator page.

In a graphical environment Muse OpenURL can be tested from Start menu `(Start/Programs/Muse/Muse OpenURL/Muse OpenURL Generator)`.

See the Muse External HTTP Server and Muse External Servlets Engine manuals, depending on the engine required, if another HTTP Web server must be used .

6.6 Starting the Muse Control Center

This is a session-oriented Web application. Once installation is complete and Java is installed on your machine (see Chapter 3, *Installing Java Virtual Machine*), check the Muse Control Center installation.

The Muse Control Center can be run under any HTTP Web server(e.g.: Apache Tomcat embedded within Muse) with servlets support.

The Muse Control Center already comes configured with a specific context for this application when it runs under the Apache Tomcat embedded within Muse.

On a successful installation, entering the address `http://localhost:8000/center/` in a Web browser should display the Muse Control Center page.

In a graphical environment the Muse Control Center can be used from Start menu `(Start/Programs/Muse/ Muse Control Center)`.

See the Muse External HTTP Server and Muse External Servlets Engine manuals, depending on the engine required, if another HTTP Web server must be used.

6.7 Starting the Muse XML DB Management System

The Muse XML DB Management System provides a method of storing XML documents in a compatible and standardized format through the XMLDB interface `( http://www.xmldb.org) .` Some Muse products use it through various APIs; for example, the Muse Personal Profiles Management System,

which is an interface for managing patron personal data .

Once installation is complete and Java is installed on your machine (see Chapter 3, *Installing Java Virtual Machine*), check the Muse XML DB Management System installation.

The Muse XML DB Management System can be run under any HTTP Web server(e.g.: Apache Tomcat embedded within Muse) with servlets support.

The Muse XML DB Management System already comes configured with a specific context for this application when it runs under the Apache Tomcat embedded within Muse.

See the Muse External HTTP Server and Muse External Servlets Engine manuals, depending on the engine required, if another HTTP Web server must be used.

7.0

Known Bugs and Issues

7.1 Running Muse behind a Proxy Server

The startup scripts used for the ICE server allow various JVM command line options to be set. You would need to adjust them only if necessary for the installation environment.

Some are network specific:

`http.proxyHost` *(default: <none>)*`http.proxyPort` *(default: 80 if* `http.proxyHost` *specified)*`http.nonProxyHosts` *(default: <none>*

`ftp.proxyHost` *(default: <none>)*`ftp.proxyPort` *(default: 80 if* `ftp.proxyHost` *specified)*`ftp.nonProxyHosts` *(default: <none>)*

`http.proxyHost` *and* `http.proxyPort` *indicate the proxy server and port that the HTTP protocol handler will use.*

`http.nonProxyHosts` *indicates the hosts which should be connected too directly and not through the proxy server. The value can be a list of hosts, each separated by a |, and in addition a wildcard character (*) *can be used for matching. For example*: `-Dhttp.nonProxyHosts="*.foo.com|localhost".`

`networkaddress.cache.ttl (default: -1)`

Specified in `java.security` *to indicate the caching policy for successful name lookups from the name service. The value is specified as integer to indicate the number of seconds to cache the successful lookup. A value of -1 indicates "cache forever".*

These properties may not be supported in future releases:

`sun.net.inetaddr.ttl`

This is a Sun private system property, which corresponds to `networkaddress.cache.ttl`. It takes the same value and has the same meaning, but can be set as a command-line option.

If the ICE Server is run as a Windows service then one should follow the steps below to add the parameters to the Java command line:

1. Stop the ICE Service from the Services panel.
2. Uninstall the ICE Service by running `${ICE_HOME}/UnInstallICEService.bat.`
3. Perform all necessary changes in `${ICE_HOME}/JavaService.jvm` file. Each line of this file represents a Java command line parameter, except the lines that start with # (these lines are considered comments). Add the following parameters to run ICE behind a proxy server (supposing the server is located on `localhost`).

-Dhttp.proxyHost=127.0.0.1

-Dhttp.proxyPort=9797

-Dftp.proxyHost=127.0.0.1

-Dftp.proxyPort=9797

1. Run the `${ICE_HOME}/InstallICEService.bat` script. This will register the ICE Service in the Windows services with the changes made.
2. Start the ICE Service from the services panel.

7.2 Upgrade Muse when Java 7 is installed

Upgrading to Muse 2509 on a machine where Java 7 is installed will take longer than upgrading on a machine with Java 6 installed. This happens because of a bug in Java 7: `BUG ID 7122740` (http://bugs.sun.com/bugdatabase/view_bug.do?bug_id=7122740). This bug is solved in Java 8 build 23. The fix is not included in the Java 7 update 1 to 4. The bug manifests when invoking a method from `java.beans.PropertyDescriptor` class which takes 5000ms in Java 7 versus 100ms in Java 6.

Note that there is not the MuseGlobal java code which calls methods from this class but these methods are intensively called by the 3rd party Installshield software tool. Also note that the upgrade is done successfully, is just that the installation will take longer.

There is no such inconvenient when performing fresh Muse install on machines with Java 7 installed.

7.3 Muse GUI Desktop Applications

It is possible to encounter display problems with GUI desktop applications from Muse with some video cards and/or operating systems. The problems manifest as very slow responses, missing repaint of the graphical interface windows, and other display anomalies. If you notice something like this in your Muse instance, please add the property `-Dsun.java2d.noddraw` to the JVM command line of the GUI Application experiencing problems. (For example, in the Search Query Translator Generator tool running under Windows, one has to edit the file `${USE_HOME}\startSQTG.bat` and alter the line starting `java` by adding the property `-Dsun.java2d.noddraw.)`