Muse Installable Components

17 January 2014

Document Version 0.0.1.1
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

Introduction

This document describes which are the Technical Components that are part of a Muse Distribution and how they are inter-related. Every Technical Installable Component presented in this document represents an entry in the Serial Number. This means the relation between the Serial Number entries and the Technical Components described here is 1:1.

The purpose of this document is to help for a better understanding of the components that are distributed according with the sales department demands.

The following chapters presents each Muse Technical Installable Component while the last chapter presents their dependency schema.

2.0

Information Connection Engine Server

The Information Connection Engine (ICE) is the core component, which provides the building blocks and the framework within which Universal Search Engine (USE) is constructed and Multi-user Universal Search Engine (Muse) exercises its control. ICE is a distributed processing system utilizing run time binding of processing modules and standardized techniques for message passing. Information Connection Engine (ICE) is the main component of the system that makes wrappers and mediators coexists. It provides access to any of them based on the client queries. It is actually more than that; it is like a small multi-user, multi-tasking operating system core. For more information please see `$MUSE_HOME/use/ice/doc/ICE Server.pdf` document.

Note: The corresponding Serial Number entry INFORMATION_CONNECTION_ENGINE_SERVER.

To see which products depend on ICE please see the Figure 1, “Muse Installable Components”.

3.0

Muse Search

The Muse Search component contains Java classes (or Java ARchive files) implementing specific ICE Processing Modules used by the Muse System.

Note: The corresponding Serial Number entry is MUSE_SEARCH.

MusSearch depends on Information Connection Engine. To see which products depend on Muse Search please see Figure 1, “Muse Installable Components”

4.0

Information Connection Engine Tools

The Information Connection Engine Tools component represents a collection of useful tools for the Muse System. Usually these tools do not need to be installed on customer machines.

These tools are:

- `PDFConverter` – this is a command line tool that provide a way to convert a PDF document to an XML Document. Currently the XML format supported by this tool is DocBook XML. The tool is written in Java, so it is independent from the platform on which it runs;
- `Muse NCIP package` – this is a command line tool that uses the NCIP protocol to execute library actions such as: request an item, create item, checking item etc.;
- `ICE Interpreter` – this is a command line tool used as an interpreter for ICE Script files. The script file has to be a valid XML file;
- `Muse Package Builder` – this is a command line tool that builds the Muse Source Packages;
- `ICE Z39.50 Client` – this is a multi-threaded Java desktop application which is part of Muse environment and allows you to open multiple simultaneous sessions with different Z39.50 servers. This application is an utility tool that a Muse administrator might used for various tests against a Z39.50 target;
- `MARC to XML Converter` – this is a command line tool that provides a flexible alternative to convert a MARC record to an XML file;
- `Muse System Information` – this is a command line tool that displays the Muse Components current version;
- `Password Encoder` – this is a command line tool that provides the user with a means of encoding passwords using either SHA1 or MD5 encryption algorithm;
- `Media Downloader` – this is a command line tool that uses as input a feed containing search results and downloads the media files specified with several rules in the configuration files;
- `ICE Client` – this is a command line client for ICE.

Note: The corresponding Serial Number entry INFORMATION_CONNECTION_ENGINE_TOOLS .

They are independent. For the other dependencies please see Figure 1, “Muse Installable Components”

5.0

Muse Web Bridge

Muse Web Bridge makes the Information Connection Engine Server functionality available through the HTTP protocol, in other words bridges between the Information Connection Engine Server API to the HTTP protocol. It enables the build of the web HTML interfaces.

This Bridge provides a way of sending HTTP requests and receiving either HTML or XML formatted responses, by means of a servlet known as `MusePeer`. Essentially is a tool which sends commands to the Information Connection Engine Server, retrieves the responses and presents them to the user in a web browser. The formatting of the interface is done using a Muse Application, which is loaded by the Muse Web Bridge. For more information please see `$MUSE_HOME/web/doc/Muse Web Bridge Communication Interface.pdf` document.

Note: The corresponding Serial Number entry is MUSE_WEB_BRIDGE.

It runs inside the Apache Tomcat Server or other External Server Engine and sends messages to Information Connection Engine Server. Muse Web Bridge also depends on Muse Search. For all dependency levels please see Figure 1, “Muse Installable Components”

5.1 Muse Web Bridge Communication Interface Kit

Muse Web Bridge Communication Interface Kit includes `testXMLAPI` tool and `Muse Web Bridge Communication Interface.pdf` manual. For more information regarding the `testXMLAPI` please see `$MUSE_HOME/web/doc/Muse Web Bridge Communication Interface.pdf` manual, section `3.2 Commands Usage HowTo`.

Note: The corresponding Serial Number entry is MUSE_WEB_BRIDGE_COMMUNICATION_INTERFACE_KIT.

It depends on Muse Web Bridge. This means that if Muse Web Bridge Communication Interface Kit is part of a Serial Number then the Muse Web Bridge will be part of that Serial Number too.

For all dependency levels please see Figure 1, “Muse Installable Components”

6.0

Muse Admin Bridge

Muse Admin Bridge is the back end used by all Administrative Consoles to perform their work. So, all the Administrative Consoles depend on Muse Admin Bridge. For more information please see `$MUSE_HOME/admin/doc/MuseAdmin Communication Interface.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_ADMIN_BRIDGE.

It runs inside the Apache Tomcat Server or other External Server Engine and depends on Muse InfoBase Bridge. Note that it fully depends on Muse Local Factory. It connects to the remote Muse Global Factory (the one that is installed on a MuseGlobal server) to retrieve some Source Package related information and store it in Muse Local Factory.

Note: Muse Global Factory must not be installed on customer machines.

Note: Muse Local Factory is usually installed on customer machines.

There are Muse Admin Bridge functionalities that depend on Information Connection Engine Tools. For example the Problem Report functionality calls Muse System Information.

For all dependency levels please see Figure 1, “Muse Installable Components”

6.1 Muse Consoles for Application Administration

The Muse Console for Application Administration (MCAA) is the tool used by Muse partner support and Implementation personnel to create and replicate Federated Search Applications for their customers and to add and update the resources that are made available for searching in Applications. For more information please see `$MUSE_HOME/admin/doc/Muse Console for Application Administration.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_CONSOLES_FOR_APPLICATIONS_ADMINISTRATION .

It fully depends on Muse Admin Bridge. For all dependency levels please see Figure 1, “Muse Installable Components”

6.2 Muse Management Console

Muse Management Console is the ultimate Administrator Console allowing configuration of every aspect of a Muse system. Usually it is not installed to customers, but only to MuseGlobal managed installations (owned, or rarely managed for customers).

Note: The corresponding Serial Number entry is MUSE_MANAGEMENT_CONSOLE .

It fully depends on Muse Admin Bridge. For all dependency levels please see Figure 1, “Muse Installable Components”

6.3 Muse Administrator Console (Standard Edition)

Muse Administrator Console (Standard Edition) allows configuration of nearly every aspect of a Muse system, including sources and users. It does not include source auditing.

Note: The corresponding Serial Number entry is MUSE_ADMINISTRATOR_CONSOLE_STANDARD.

It fully depends on Muse Admin Bridge. For all dependency levels please see Figure 1, “Muse Installable Components”

6.4 Muse Admin Bridge Communication Interface Kit

Muse Admin Bridge Communication Interface Kit includes `MuseAdmin Client` tool and `Muse Admin Bridge Communication Interface.pdf` manual. `MuseAdmin` Client is a tool that connects to a Muse Admin via TCP/IP and executes `MuseAdmin` commands listed in an XML document received as input. For more information regarding this tool please see `$MUSE_HOME/admin/tools/client/doc/MuseAdmin Client.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_ADMIN_BRIDGE_COMMUNICATION_INTERFACE_KIT.

It depends on Muse Admin Bridge. This means that, if Muse Admin Bridge Communication Interface Kit is part of a Serial Number then the Muse Admin Bridge will be part of that serial Number, too.

For all dependency levels please see Figure 1, “Muse Installable Components”

7.0

Muse Z39.50 Bridge

Muse Z39.50 Bridge acts as a gateway to a collection of heterogeneous sources like, Z39.50 Servers, HTTP, Telnet or SQL data sources. For more information please see `$MUSE_HOME/z3950/doc/Muse Z39.50 Bridge.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_Z3950_BRIDGE.

It runs inside the Apache Tomcat Server or other External Server Engine and sends messages to Information Connection Engine Server. For all dependency levels please see Figure 1, “Muse Installable Components”

8.0

Muse SOAP Bridge

The Muse SOAP Bridge provides an Endpoint for exposing the search capabilities of Muse as SOAP services. Once the connection between the client and the server has been established the client sends a SOAP request, the server processes it and sends back a SOAP response. Both the request and the response must be XML documents conforming to the SOAP encoding. For more information please see `$MUSE_HOME/soap/doc/Muse SOAP Bridge.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_SOAP_BRIDGE.

It runs inside the Apache Tomcat Server or other External Server Engine and sends messages to Information Connection Engine Server. For all dependency levels please see Figure 1, “Muse Installable Components”

9.0

Muse Proxy Server

Starting with Muse 2.6.0.0 Muse Proxy Server doesn't come with Muse distribution and will not be installed using Muse Setup. Muse Proxy Server is an HTTP proxy with filtering capabilities. Implemented filters perform various tasks needed in the Muse system (such as page rewriting, in the Navigation Manager filter). For more information please see `$MUSE_HOME/proxy/doc/Muse Proxy.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_PROXY_SERVER.

For all dependency levels please see Figure 1, “Muse Installable Components”

9.1 Muse Proxy Applications

A Muse Proxy Application is a fully configurable interface, which supports single point access control where users are authenticated to a variety of subscribed services. A subscribed service is modeled in a Muse Proxy Application through a source. In this way, Muse Proxy protects the source credentials, using the federated approach, while leaving the user to access the source target site in an authenticated manner.

Note: The corresponding Serial Number entry is MUSE_PROXY_APPLICATIONS.

It runs inside Muse Proxy Server. For all dependency levels please see Figure 1, “Muse Installable Components”

10.0

Apache Tomcat embedded within Muse

Apache Tomcat Server 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.

Note: The corresponding Serial Number entry is APACHE_TOMCAT.

11.0

Muse HTTP Server

Muse HTTP Server is an HTTP/1.0 compatible server for making hypertext and other documents available to Web browsers, supporting Servlet 2.2 and JSP 1.1 specifications. All bridges are running inside the HTTP Server. For more information please see `$MUSE_HOME/http/doc/Muse HTTP Server.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_HTTP_SERVER.

Note: This product is deprecated and Apache Tomcat must be used instead.

For all dependency levels please see Figure 1, “Muse Installable Components”

12.0

Muse Enrichment Service

Muse Enrichment Service provides means for "content enrichment", that is to supplement bibliographic records, e.g. with TOC, book reviews or cover images generated by dynamic look ups in third party data repositories.

Note: The corresponding Serial Number entry is MUSE_ENRICHMENT_SERVICE.

It runs inside the Apache Tomcat Server or other External Server Engine. For all dependency levels please see Figure 1, “Muse Installable Components”

13.0

Muse OpenURL

The OpenURL is a protocol for interoperability between an information resource and a service component that offers localized services in an open linking environment. For more information please see `$MUSE_HOME/openurl/doc/Muse OpenURL.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_OPEN_URL.

It runs inside the Apache Tomcat Server or other External Server Engine. For all dependency levels please see Figure 1, “Muse Installable Components”

14.0

Muse Web2 Bridge

Muse Web2 Bridge is not a standalone program but a connector running under a complex framework environment. It uses the embedded Apache Tomcat Server that is to some extent a meta server or it can use any other HTTP server with at least servlets 2.1 compliant support. For more information please see `$MUSE_HOME/web2/doc/Muse Web2 Bridge.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_WEB2_BRIDGE.

It runs inside the Apache Tomcat Server or other External Server Engine and sends messages to Information Connection Engine Server.

This is a custom partner bridge.

For all dependency levels please see Figure 1, “Muse Installable Components”

15.0

Muse Serial Number Encoder

Muse Serial Number Encoder is used to generate Serial Numbers and can extend them using Muse Registration Service. It uses the Muse File Encoder tool to encrypt/decrypt a file using DES, DESede (3DES) algorithms.

It is to be used for MuseGlobal staff only.

For more information please see `$MUSE_HOME/use/tools/serial/doc/Muse Serial Number Encoder.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_SERIAL_ENCODER.

For the dependencies please see Figure 1, “Muse Installable Components”

16.0

Muse InfoBase Bridge

Muse InfoBase Bridge provides a layer for accessing the InfoBase database (including Source information). It is an essential component (together with Muse InfoBase Management System - IBMS) if Source Factory access is desired in the Muse installation. For more information please see `$MUSE_HOME/factory/doc/Muse Source Factory Definitions and Guide.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_INFOBASE_BRIDGE.

It depends on Muse XMLDB Management System. For all dependency levels please see Figure 1, "Muse Installable Components"

16.1 Muse Local Source Factory

Muse Local Source Factory is the local component of the Source Factory. If Source Management functionality is desired, then this component is mandatory.

Note: The corresponding Serial Number entry is MUSE_LOCAL_SOURCE_FACTORY.

It depends on Muse InfoBase Bridge. For all dependency levels please see Figure 1, "Muse Installable Components"

16.2 Muse Global Source Factory

Muse Global Source Factory is the component that is installed only on the designated MuseGlobal server(s). It is not to be installed in customer installation.

Note: The corresponding Serial Number entry is MUSE_GLOBAL_SOURCE_FACTORY.

It depends on Muse InfoBase Bridge. For all dependency levels please see Figure 1, "Muse Installable Components"

16.3 Muse Partner Source Factory

Muse Partner Source Factory is the component that is installed only on the designated partner server(s). If Muse Global Source Factory is installed then Muse Partner Source Factory mus not be installed. All source database records: `DAR(Data Administration Record), HAR(Host Administration Record), SAR(Source Administration Record), TAR(Target Administration Record), AAR(Authenticator Administration Record), CAR(Connector Ad ministration Record)` corresponding to that partner are exported from Muse Global Source Factory and imported into Partner Source Factory.

Note: The corresponding Serial Number entry is MUSE_PARTNER_SOURCE_FACTORY.

It depends on Muse InfoBase Bridge. For all dependency levels please see Figure 1, “Muse Installable Components”

17.0

Muse Control Center

Muse Control Center is a utility tool used by the administrator of a Muse environment and is part of the Muse package. It is used to test various components of the Muse environment or performed certain actions based on scheduled tasks. For more information please see `$MUSE_HOME/center/doc/Muse Control Center.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_CONTROL_CENTER.

It is independent. For all the other dependency levels please see Figure 1, “Muse Installable Components”

18.0

Muse Builder

Muse Builder is a tool that automates many of the redundant activities of the Muse Sources and Muse Application Web Interfaces development. Automating this activity would lead to a better management of the Muse Application Web Interfaces and Muse Sources department, less development time and of course better support for customers.

Muse Builder is a framework, where each integrated tool are placed together for allowing developers to easily develop Muse Connectors and Muse Application Web Interfaces. It is only the specially configured tools that makes Muse Builder a designated Muse utility tool, other way it might be a generic application to handle and edit various resources logically grouped under projects.

Generally it is not installed on a customer installation. For more information please see `$MUSE_HOME/builder/doc/Muse Builder.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_BUILDER.

For the dependencies please see Figure 1, “Muse Installable Components”

18.1 Muse Connectors Generator

Muse Connectors Generator is an essential tool used by the developers in writing specialized HTTP wrappers (connectors).

Generally it is not installed on a customer installation.

For more information please see `$MUSE_HOME/generator/doc/Muse Connectors Generator.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_CONNECTORS_GENERATOR.

It depends on Muse Builder and Muse Search. For more dependencies please see Figure 1, “Muse Installable Components”

18.2 Muse Search Query Translator Generator

The Muse system uses the XML format to internally represent a query. Muse defines the Internal Search Representation (ISR) in XML structured format to describe a query. The ISR generated from the user's query gets converted by a XSL translator into a query string to be passed over to the Data Source. Because the XSL translators are difficult to write we have decided to create a Data Source Description (DSD) which can describe each Data Source and have the XSL translators automatically generated based on the DSD files. These description files are created with the Structured Query Translator Generator (SQTG) utility tool.

Generally it is not installed on a customer installation.

For more information please see `$MUSE_HOME/sqtg/doc/Muse SQTG.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_SEARCH_QUERY_TRANSLATOR_GENERATOR.

It depends on Muse Builder and Muse Search. For more dependencies please see Figure 1, “Muse Installable Components”

18.3 Muse Source Package Testing

Muse Source Package Testing is an internal MuseGlobal tool used to test Source Packages, usually during the development phase. It is not to be installed with customer installations.

Generally it is not installed on a customer installation.

For more information please see `$MUSE_HOME/use/tools/sptesting/doc/SPTesting.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_SOURCE_PACKAGE_TESTING.

It depends on Muse Builder and Muse Search. For more dependencies please see Figure 1, “Muse Installable Components”

18.4 Muse Source Package Assistant

Muse Source Package Assistant is an internal MuseGlobal tool used to help programmers in the process of Source Packages development. It is not to be installed with customer installations.

Generally it is not installed on a customer installation.

For more information please see `$MUSE_HOME/spassistant/doc/Muse Source Package Assistant` manual.

Note: The corresponding Serial Number entry is MUSE_SOURCE_PACKAGE_ASSISTANT.

It depends on Muse Builder and Muse Search. For more dependencies please see Figure 1, "Muse Installable Components"

19.0

Muse XML BD Management System

Muse XML DB Management System provides the core components for accessing an XML database. Several other components (GAMS, PPMS, PWMS, etc.) exist that make use of this layer. For more information please see `$MUSE_HOME/xmldb/doc/Muse XMLDB.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_XMLDB_MANAGEMENT_SYSTEM.

For all dependency levels please see Figure 1, "Muse Installable Components"

19.1 Muse Personal Profiles Management System

Muse Personal Profiles Management System is a component that allows user profiles to be stored in an XML database. The profiles can afterwards be read/modified.

Note: The corresponding Serial Number entry is MUSE_PERSONAL_PROFILES_MANAGEMENT_SYSTEM.

It depends on Muse XML BD Management System. It is also a slight functional dependency between Muse Personal Profiles Management System and Muse Personal WorkRoom Management System and is recommended to be both installed on customer machine.

For all dependency levels please see Figure 1, "Muse Installable Components"

19.2 Muse Personal WorkRoom Management System

Muse Personal WorkRoom Management System is a component that allows user WorkRoom to be stored in an XML database.

Note: The corresponding Serial Number entry is

MUSE_PERSONAL_WORKROOM_MANAGEMENT_SYSTEM.

It depends on Muse XML BD Management System. It is also a slight functional dependency between Muse Personal Profiles Management System and Muse Personal WorkRoom Management System and is recommended to be both installed on customer machine.

For all dependency levels please see Figure 1, “Muse Installable Components”

19.3 Muse Global Authorizing Management System

Muse Global Authorizing Management System can provide single signon functionality for several Muse components. IT is not installed on customer installation.

Note: The corresponding Serial Number entry is MUSE_GLOBAL_AUTHORIZING_MANAGEMENT_SYSYEM.

It depends on Muse XML BD Management System. For all dependency levels please see Figure 1, “Muse Installable Components”

19.4 Muse InfoBase Management System

Muse InfoBase Management System provides a layer for accessing the InfoBase database (including Source information). It is an essential component (together with Muse InfoBase Bridge) if Source Factory access is desired in the Muse installation.

Note: The corresponding Serial Number entry is MUSE_INFOBASE_MANAGEMENT_SYSTEM.

It depends on Muse XML BD Management System. For all dependency levels please see Figure 1, “Muse Installable Components”

20.0

Muse Statistics Monitor

Muse Statistics Monitor is an interactive monitoring tool for the administrator of a Muse environment. Its function is to present various information of Muse system usage out of the log files. For more information please see `$MUSE_HOME/monitor/doc/Muse Statistics Monitor.pdf` manual.

Muse Statistics Monitor includes Muse JMX Monitor, a tool that polls values exported by ICE Server through JMX. Muse JMX Monitor was designed so that it could connect to multiple ICE Servers and allow customized data polling. The data for each server will be stored in RRD files, using the JRobin API. Muse JMX Monitor will handle only the data polling and storing part, while the graphing will be covered by Muse RRD Grapher or a customized tool that will analyze the RRD files created by Muse JMX Monitor.

Note: The corresponding Serial Number entry is MUSE_STATISTICS_MONITOR.

It is independent. For all dependency levels please see Figure 1, “Muse Installable Components”

21.0

Muse AAA Bridge

Muse AAA Bridge is not a standalone program but a connector running under a complex framework environment. It uses the embedded Apache Tomcat Server that is to some extent a meta server or it can use any other HTTP server with at least servlets 2.1 compliant support.

This is a custom partner bridge, AAA is the partner code.

For more information please see `$MUSE_HOME/AAA/doc/Muse AAA Bridge.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_AAA_BRIDGE.

It runs inside the Apache Tomcat Server or other External Server Engine and sends messages to Information Connection Engine Server. For all dependency levels please see Figure 1, “Muse Installable Components”

22.0

Muse One Box

Muse is an external provider for the Google OneBox Appliance. Connecting Muse and OneBox is carried out by means of a special bridge, called Muse OneBox Bridge. For more information please see `$MUSE_HOME/onebox/doc/Muse OneBox Bridge.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_ONE_BOX .

Initially this was built for development demo purposes.

It runs inside the Apache Tomcat Server or other External Server Engine and sends messages to Information Connection Engine Server. For all dependency levels please see Figure 1, “Muse Installable Components”

23.0

Muse Content Mining

Content Mining is the process of extracting "meaning" from objects and representing that meaning in a standardised, structured manner. It is also the general name given within MuseGlobal to the whole process of creating, managing, manipulating and displaying meaning, from wherever it is derived. For more information please see `$MUSE_HOME/doc/Content Mining in Muse.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_CONTENT_MINING.

It depends on Muse Search. For all dependency levels please see Figure 1, “Muse Installable Components”

24.0

Muse SES Bridge

The purpose of Muse SES Bridge is to mimic the behavior of an Oracle SES instance and to provide some of the services an Oracle SES instance does. Muse SES Bridge can be used as a federated source for an Oracle SES instance. For more information please see `$MUSE_HOME/ses/doc/Muse SES Bridge.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_SES_BRIDGE.

Initially this was built for development demo purposes.

It runs inside the Apache Tomcat Server or other External Server Engine and sends messages to Information Connection Engine Server. For all dependency levels please see Figure 1, “Muse Installable Components”

25.0

Muse LEXS Bridge

The Muse LEXS Bridge is a SOAP based protocol bridge. The LEXS protocol defines two types of operations: Publication and Discovery (PD) and Search and Retrieval (SR). The Muse LEXS Bridge implements only the SR operation type the other operation type being not appropriate for Muse as at the current time. For more information please see `$MUSE_HOME/lexs/doc/Muse LEXS Bridge.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_LEXS_BRIDGE.

Initially this was built for development demo purposes.

It runs inside the Apache Tomcat Server or other External Server Engine and sends messages to Information Connection Engine Server. For all dependency levels please see Figure 1, “Muse Installable Components”

26.0

Muse DPS NetWeaver Enterprise Search Bridge

The Muse DPS NetWeaver Enterprise Search provides two NWES DPS Enpoints through which Muse can export data to Netweaver Enterprise Search. The communication protocol is based on SOAP. For more information please see `$MUSE_HOME/nwes/doc/Muse NWES Bridge.pdf` manual.

Note: The corresponding Serial Number entry is MUSE_NWES_BRIDGE.

Initially this was built for development demo purposes.

It runs inside the Apache Tomcat Server or other External Server Engine and sends messages to Information Connection Engine Server. For all dependency levels please see Figure 1, “Muse Installable Components”

27.0

Muse Record Tracking System

The Muse Record Tracking system is intended to be used in Harvesting type Muse installations. Its main role is to track the statuses of the records processed by the system from the moment these records enter the system either through a search connector or imported from 3rd party databases to the moment these records are exported by the system in one of the various outputs.

Note: The corresponding Serial Number entry is MUSE_RECORD_TRACKING_SYSTEM.

28.0

Muse Document Repository

The Muse Document Repository is used to store documents. The Muse Document Repository is implemented using the `Java Content Repository (JSR-170 and JSR-283)which` is a set of interfaces which provide an API for using the Content Repository. The Muse Document Repository has a collection of workspaces, one for each of the ICE individual users so that each user sees only his own documents. Once the documents are stored in the Muse Document Repository they will be deleted when the user signs out.

Note: The corresponding Serial Number entry is MUSE_DOCUMENT_REPOSITORY.

29.0

Muse Central Index

Muse Central Index is a metadata repository that can be searched by Muse Information Discovery. The record format used by Muse Central Index is a Dublin Core-based schema. Muse Central Index can index e-book and article metadata, catalog records, and other information harvested from institutional re positories and other digital collections via the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH). Muse Central Index is not pre-populated with metadata but each installation can add its own sources to build an actual index.

Note: The corresponding Serial Number entry is MUSE_CENTRAL_INDEX.

Make sure you have a working Muse installation that contains a harvesting application and an application to search Muse Central Index. In order to harvest and index records, you need to start Muse Control Center. For all dependency levels please see Figure 1, “Muse Installable Components”

30.0

Muse Applications

A Muse Application is an individual instance of the Muse program that runs at a customer site or at a group of sites to provide federated searching or harvesting of resources.

It depends on Information Connection Engine, Muse Search and Muse Web Bridge. A Muse Application does not exist if its corresponding ICE user is not created. Also a Muse Application contains the Muse Connectors and Muse Modules (that are part of Muse Search) that permit the search and the pre and post processing. The build of the Muse Application web HTML interface is enabled by Muse Web Bridge.

For more dependencies please see Figure 1, “Muse Installable Components”

31.0

Interdependence

The schema below shows the interdependence between Muse installable components.

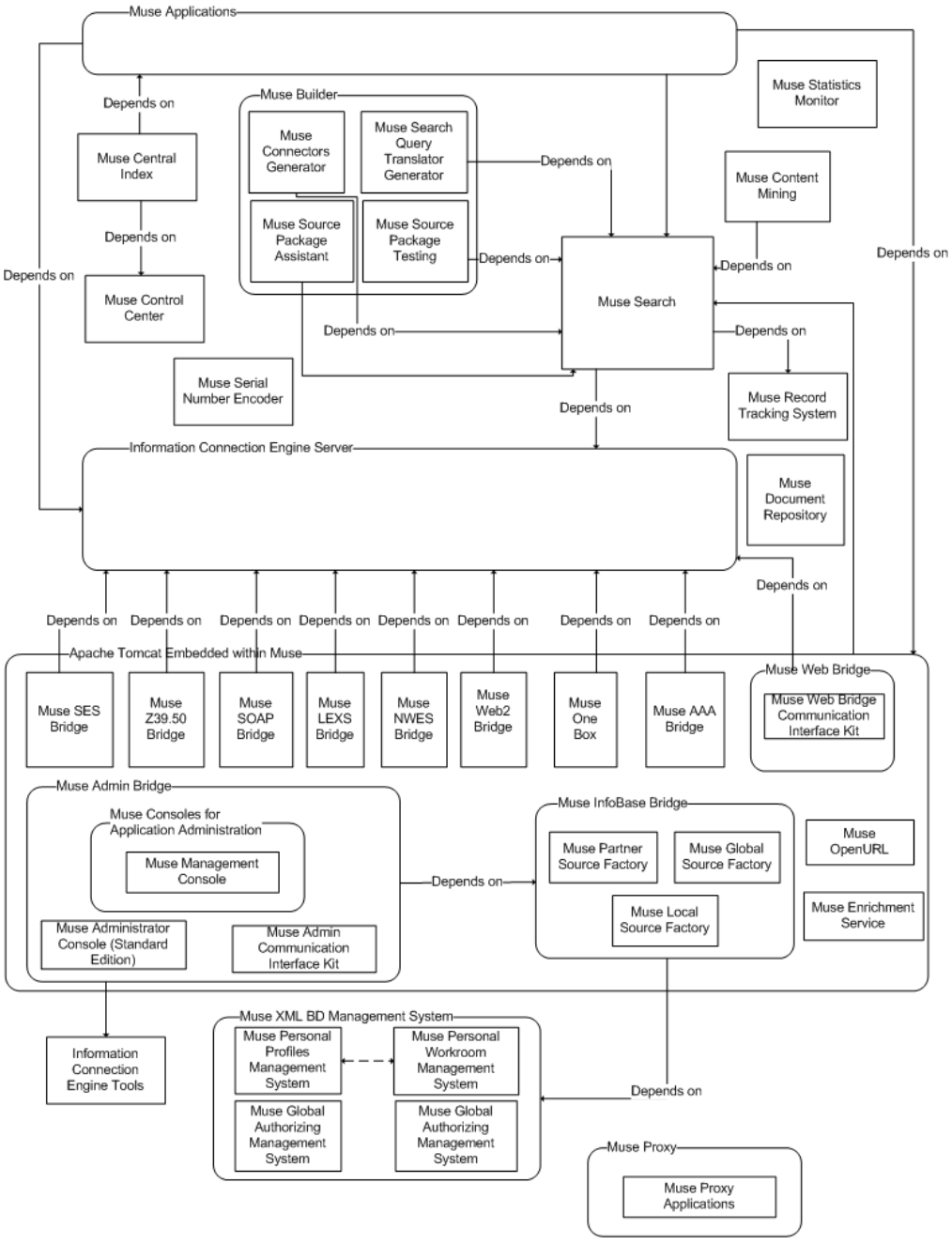

Figure 1. Muse Installable Components

32.0

Summary

The aim of this chapter is to provide a quick and easy guide for Customer Support on Products to enable for Customer Serial Numbers.

Numbers refer to sections/chapters , where additional description and detail may be found.

Chapter 2, *Information Connection Engine Server*	**Information Connection Engine Server**	**Always included**
Chapter 3, *Muse Search*	**Muse Search**	**Always included**
Chapter 4, *Information Connection Engine Tools*	Information Connection Engine Tools	Not normally included for customers. If an individual tool is necessary for a Product it is installed by enabling that Product
Chapter 5, *Muse Web Bridge*	**Muse Web Bridge**	**Normally included**
Section 5.1, "Muse Web Bridge Communication Interface Kit"	*- Muse Web Bridge Communication Interface Kit*	*Only included if a customer is licensed to write to the Muse Web Bridge*
Chapter 6, *Muse Admin Bridge*	**Muse Admin Bridge**	**Normally included**
Section 6.1, "Muse Consoles for Application Administration"	**- Muse Consoles for Application Administration**	**Normally included unless specifically not licensed**
Section 6.2, "Muse Management Console"	- Muse Management Console	Only installed on MuseGlobal servers
Section 6.3, "Muse Administrator Console (Standard Edition)"	*- Muse Administrator Console (Standard Edition)*	*Normally not included*
Section 6.4, "Muse Admin Bridge Communication Interface Kit"	*- Muse Admin Bridge Com munication Interface Kit*	*Only included if a customer is licensed to write to the Muse Admin Bridge*
Chapter 7, *Muse Z39.50 Bridge*	Muse Z39.50 Bridge	Only installed if licensed
Chapter 8, *Muse SOAP Bridge*	Muse SOAP Bridge	Only installed if licensed
Chapter 9, *Muse Proxy Server*	**Muse Proxy Server**	**Normally included unless specifically not licensed**
Chapter 10, *Apache Tomcat embedded within Muse*	**Apache Tomcat Embedded within Muse**	**Always included**

Chapter 11, *Muse HTTP Server*	Muse HTTP Server	Not normally included. It si now deprecated.
Chapter 12, *Muse Enrichment Service*	Muse Enrichment Service	Not currently included
Chapter 13, *Muse OpenURL*	Muse OpenURL	Not currently included
Chapter 14, *Muse Web2 Bridge*	Muse Web2 Bridge	Not normally included – Partner specific
Chapter 15, *Muse Serial Number Encoder*	Muse Serial Number Encoder	MuseGlobal licenses only
Chapter 16, *Muse InfoBase Bridge*	**Muse InfoBase Bridge**	**Normally included**
Section 16.1, "Muse Local Source Factory"	**- Muse Local Source Factory**	**Normally included**
Section 16.2, "Muse Global Source Factory"	- Muse Global Source Factory	MuseGlobal licenses only
Section 16.3, "Muse Partner Source Factory"	*- Muse Partner Source Factory*	*Only included if partner has licensed Partner Source Factory*
Chapter 17, *Muse Control Center*	**Muse Control Center**	**Normally included unless specifically not licensed**
Chapter 18, *Muse Builder*	*Muse Builder*	*Normally only MuseGlobal licenses, unless Partner Source Factory is licensed*
Section 18.1, "Muse Connectors Generator"	*- Muse Connectors Generator*	*Normally only MuseGlobal licenses, unless Partner Source Factory is licensed*
Section 18.2, "Muse Search Query Translator Generator"	*- Muse Search Query Translator Generator*	*Normally only MuseGlobal licenses, unless Partner Source Factory is licensed*
Section 18.3, "Muse Source Package Testing"	*- Muse Source Package Testing*	*Normally only MuseGlobal licenses, unless Partner Source Factory is licensed*
Section 18.4, "Muse Source Package Assistant"	*- Muse Source Package Assistant*	*Normally only MuseGlobal licenses, unless Partner Source Factory is licensed*
Chapter 19, *Muse XML BD Management System*	**Muse XML DB Management System**	**Normally included**
Section 19.1, "Muse Personal Profiles Management System"	*- Muse Personal Profiles Management System*	*Only installed if licensed*
Section 19.2, "Muse Personal WorkRoom Management System"	*- Muse Personal WorkRoom Management System*	*Only installed if licensed*
Section 19.3, "Muse Global	- Muse Global Authorizing	Not currently included

Authorizing Management System"	Management System	
Section 19.4, "Muse InfoBase Management System"	**- Muse InfoBase Management System**	**Normally included unless specifically not licensed**
Chapter 20, *Muse Statistics Monitor*	**Muse Statistics Monitor**	**Normally included unless specifically not licensed**
Chapter 21, *Muse AAA Bridge*	*Muse AAA Bridge*	*Not normally included – Partner specific*
Chapter 22, *Muse One Box*	Muse One Box	Not normally included
Chapter 23, *Muse Content Mining*	*Muse Content Mining*	*Only installed if licensed*
Chapter 24, *Muse SES Bridge*	Muse SES Bridge	Not normally included
Chapter 25, *Muse LEXS Bridge*	Muse LEXS Bridge	Not normally included
Chapter 26, *Muse DPS NetWeaver Enterprise Search Bridge*	Muse DPS Netweaver Enterprise Search Bridge	Not normally included
Chapter 27, *Muse Record Tracking System*	*Muse Record Tracking System*	*Only installed if licensed*
Chapter 28, *Muse Document Repository*	*Muse Document Repository*	*Only installed if licensed*
Chapter 29, *Muse Central Index*	*Muse Central Index*	*Only installed if licensed*

Table 1.