Release Notes for Sun GlassFish ESB v2.2

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 821–0857–10
December 2009

100106@23031

Contents

Release Notes for Sun GlassFish ESB v2.2

GlassFish ESB delivers a lightweight and agile ESB platform that packages the innovation happening with Project OpenESB into a commercially supported, enterprise-class platform. GlassFish ESB builds on over 18 years of Sun expertise within SOA, ESB, integration, and server-side Java platforms. In essence, GlassFish ESB is a binary distribution that combines technology from Project OpenESB, the GlassFish application server, and the NetBeans IDE communities into a supported, commercial distribution. GlassFish ESB is compatible with a number of JBI-compliant plug-ins that provide a range of different technical capabilities.

This topic provides information about changes made to GlassFish ESB between versions 2.1 and 2.2. If you have any questions, go to the GlassFish ESB web site at `http://www.glassfish-esb.org`.

To access all the GlassFish ESB documentation, go to the GlassFish ESB documentation page at `http://docs.sun.com/app/docs/coll/2511.2?l-en`.

What You Need To Know

These release notes include the following topics:

- "What's New in GlassFish ESB 2.2" on page 7
- "Known Issues for GlassFish ESB" on page 11

GlassFish ESB Supported Operating Systems and External Systems

GlassFish ESB is divided into design-time components, which run on the NetBeans IDE, and runtime components, which run on the GlassFish application server. Design-time and runtime components each support a specific set of operating systems. In addition, several GlassFish components support external systems, such as database platforms, LDAP servers, JMS servers, and so on.

For a complete list of supported operating systems and external systems, see "GlassFish ESB Supported Operating Systems and External Systems" in *Planning for GlassFish ESB Installation*.

GlassFish ESB Components

The standard GlassFish ESB installation includes several service engines, binding components, editors, and tools to help you create your JBI applications. You can purchase additional packages to extend the functionality of your basic GlassFish ESB environment.

The following topics describe what is installed for each type of installation:

- "GlassFish ESB Standard Installation" on page 6
- "GlassFish ESB Platinum Pack Installation" on page 7
- "GlassFish ESB Healthcare Pack Installation" on page 7

GlassFish ESB Standard Installation

A standard GlassFish ESB installation includes the following products:

Service Engines

- BPEL
- Data Mashup
- Java EE
- POJO
- XSLT

Binding Components and Encoders

- Database
- Email
- File
- FTP
- HTTP
- JMS (includes JMS JCA)
- LDAP
- REST
- Scheduler

Editors and Project Types

- CASA Editor
- WSDL Editor
- Composite Application

GlassFish ESB Platinum Pack Installation

The GlassFish ESB Platinum Pack extends the functionality and features of GlassFish ESB by adding the components listed below.

- IEP Service Engine
- Worklist Manager Service Engine
- COBOL Copybook Encoder
- BPEL Monitor Console
- Event Management API

GlassFish ESB Healthcare Pack Installation

The GlassFish ESB Healthcare Pack extends the functionality and features of GlassFish ESB by adding the components listed below.

- Sun Master Index
- HL7 Binding Component
- PIX/PDQ Manager

What's New in GlassFish ESB 2.2

New Supported Platforms

- NetBeans IDE 6.7.1
- GlassFish Enterprise Server 2.1.1
- Microsoft Windows Server 2008 (32 bit)
- Red Hat Linux Server 5.3 Basic Server and Advanced Platform (32 and 64 bit)
- IBM AIX 5L Version 6.1 (64 bit) (with IBM's JDK 6.1 SR7)
- Mozilla Firefox 3.5.2 and Safari 3.2 for accessing all GlassFish ESB browser components

New GlassFish ESB Components

GlassFish ESB includes the following new components in version 2.2:

- **Sun Master Index** – A customizable framework that allows you to create master patient indexes to uniquely identify and cross-reference patients and other participants whose information is stored in disparate systems throughout your organization.
- **PIX/PDQ Manager** – A group of applications that address integration issues specific to sharing healthcare information based on the guidelines put forth by Integrating the Healthcare Enterprise (IHE). The applications form a specialized implementation of HL7 messaging that facilitates the creation of an Information Health Exchange.
- **HL7 Binding Component** – A binding component that allows GlassFish ESB application to establish and maintain connections with HL7 v2 messaging systems, manage message enveloping and routing, and perform message validations.
- **COBOL Copybook Encoder** – A fixed-length field-based encoder that provides the tools you need to define and test COBOL Copybook encoders as well as provide runtime support for the encoders.
- **Email Binding Component** – A binding component that supports configuration and connection to email servers within a JBI environment, allowing emails to be sent from and received by a JBI application.
- **REST Binding Component** – A binding component that provides external connectivity for REST over HTTP from JBI applications, which allows external systems to invoke RESTful web services hosted by the JBI platform and allows JBI components to invoke external RESTful web services.
- **POJO Service Engine** – A service engine that allows you to define business logic in Java (as Plain Old Java Objects) and then deploy those business objects in a JBI component, providing a simple and flexible method for using Java in integration applications.
- **Worklist Manager Service Engine** – A service engine that allows you to define manual tasks to be inserted into an otherwise automated workflow, such as a BPEL process, and to then manage and monitor the handling of any generated tasks.
- **BPEL Monitor** – A monitoring tool for the BPEL Service Engine's application and processes that provides a comprehensive view of active applications in real-time, allowing you to quickly ascertain the health of the system.
- **Event Management** – API support for retrieving alerts from JBI runtime components and applications that push alerts to subscribed clients through reverse-ajax.

BPEL Service Engine Enhancements

The following enhancements were made to the BPEL SE for this release:

- Exceptions defined in referenced resources can be selected from a fault name selector instead of needing to be typed in manually (NetBeans Issue 173313).
- The BPEL SE can initiate a transaction context (TX) even when the BPEL SE does not have persistence enabled. To support this, a new optional attribute named `atomicTxType` was added to the BPEL `process` element (NetBeans Issue 169664).
- A new Java function in the BPEL menu of the Mapper allows you to call the classes and methods of your project JAR files and referenced resources.
- Validation for the `doXslTranform` function checks that the specified XSL file can be found (NetBeans Issue 163856).
- You can navigate to EJB modules from partner links in a BPEL process (NetBeans Issue 124684).
- Clicking on a referenced resource node brings up a dialog box that allows you to edit the resource (NetBeans Issue 164161).
- Application variables are supported with the BPEL SE (Issue 1251).
- BPEL process can be dynamically invoked directly from another BPEL process without going through the HTTP BC (Issue 2060).
- The BPEL SE has graphical support for configuring Normalized Message properties.

IEP Service Engine Enhancements

The following enhancements were made to the IEP SE for this release:

- Supports MySQL Enterprise Server 5.1.

File Binding Component Enhancements

The following enhancements were made to the File BC for this release:

- Allows you to poll subdirectories, and provides a regular expression-based exclude mechanism.
- On-demand read is now supports file name patterns, regular expressions, and reading subdirectories. The expression-based exclude mechanism is available for on-demand reads as well.

FTP Binding Component Enhancements

The following enhancements were made to the FTP BC for this release:

- Supports database-based synchronization for clustering. Previously, only file-based synchronization could be was available.

HTTP Binding Component Enhancements

The following enhancements were made to the HTTP BC for this release:

- Allows you to insert application logic to change the protocol or payload messages before or after calling a service using JAX-WS handlers.
- Resolves multiple host names by allowing you to specify a list of alias names used to validate the hostname in an HTTP URL. This adds a new runtime property, `Allowed Hostname (Aliases) for localhost`.
- Supports connection timeout, read timeout, and hostname verifier settings for HTTP binding requests.
- Provides message tracking and nested diagnostic context logging support.
- Allows ?wsdl queries to be disabled through a CASA/binding configuration.
- Supports optional SOAP header propagation.

JMS Binding Component Enhancements

The following enhancements were made to the JMS BC for this release:

- Supports request/reply using BPEL correlation.
- The JMS JCA provides an option for interceptors, which adds the capability to the JMS BC. These can be defined globally or for a specific instance.

Scheduler Binding Component Enhancements

The following enhancements were made to the Scheduler BC for this release:

- Supports Catalog Resolver so WSDL files can be dragged into a BPEL project from a different project without exceptions (Issue 1969).
- Supports the Redelivery On-Failure Suspend option (Issue 2098).

Issues Addressed in This Release

For links to complete lists of the fixes that went into GlassFish ESB, go to Issues Addressed in This Release (`http://wiki.open-esb.java.net/Wiki.jsp?page=GFESBReleaseNotesv2.2#section-GFESBReleaseNotesv2.2-IssuesAddressedInThi`
This page provides a link to the following:

- Runtime Fixes
- Design-Time Fixes
- Additional Data Mashup Fixes

Known Issues for GlassFish ESB

This section lists known issues for this release along with workarounds when available. Issues are listed for the following components:

- "BPEL Service Engine" on page 12
- "BPEL Monitoring Console" on page 13
- "COBOL Copybook Encoder" on page 13
- "Data Mashup Service Engine" on page 13
- "Database Binding Component" on page 14
- "Email Binding Component" on page 16
- "File Binding Component" on page 16
- "FTP Binding Component" on page 16
- "HL7 Binding Component" on page 17
- "IEP Service Engine" on page 17
- "Java EE Service Engine" on page 19
- "Java Application Project for JAXB" on page 19
- "LDAP Binding Component" on page 19
- "PIX/PDQ Manager" on page 19
- "REST Binding Component" on page 20
- "Worklist Manager Service Engine" on page 20
- "Sun Master Index" on page 20
- "XSLT Service Engine" on page 20
- "Cluster Configuration" on page 21
- "Composite Application" on page 21
- "Installation and Upgrading" on page 22
- "JBI Management" on page 23
- "Localization" on page 23

BPEL Service Engine

Note – For BPEL SE stop/restart issues, see "JBI Management" on page 23.

The following are known issues for the BPEL Service Engine:

- NetBeans Issue 148288 (`http://www.netbeans.org/issues/show_bug.cgi?id=148288`)

  If a BPEL project is deleted using the Delete Source option and a new project is then created that reuses the same name, the newly created project's Properties window cannot be opened.

  To work around this issue, stop and then restart the NetBeans IDE.
- Issue 555 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=555`)

  The BPEL Service Engine only creates connection pools and data sources for the Derby database. This functionality is available to assist developers and is not available for other databases or in a cluster. The BPEL Service Engine does create the persistence tables automatically for all supported databases, but only in standalone mode; this feature is not available in cluster mode. If you create connection pools, data sources, and tables before you install the BPEL Service Engine, it is possible to install the component with persistence enabled. Otherwise, installing to the cluster will fail.

  To work around the issue of tables not being automatically created, you can create the tables in standalone mode and then add the BPEL SE to the cluster. For more information, see "Setting Up the BPEL Database" in *Configuring GlassFish ESB for Clustering*.
- NetBeans Issue 178853 (`http://www.netbeans.org/issues/show_bug.cgi?id=178853`)

  When using the new feature that allows you to map a partner link role node to a variable or EndpointReferenceType node, the generated `pseudoComp` element has an incorrect namespace defined in its `type` and `qName` attributes.

  To work around this issue, modify the namespace prefixes for `type` and `qName` to the prefix for `http://schemas.xmlsoap.org/ws/2004/08/addressing`.
- Issue 2592 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=2592`)

  When you configure a BPEL process to send a callback address, it involves copying a partner link value to a variable. If this address is intended for a service that is not within the same JVM, the associated partner link port type, messages, binding, and services all need to be defined in one WSDL document.
- Issue 2681 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=2681`)

  Failover does not work correctly when validation is enabled for the BPEL Service Engine.

  To work around this issue, disable validation in the service engine runtime properties.
- NetBeans Issue 167638 (`http://www.netbeans.org/issues/show_bug.cgi?id=167638`)

  If you save changes on the JavaScript Editor using the Save button, an IllegalStateException occurs.

To work around this issue, save the BPEL process after any changes are made on the JavaScript Editor.

- Deployment fails for BPEL projects if the BPEL Service Engine has monitoring enabled but has persistence disabled in its runtime properties.

  To work around this issue, enable persistence for the BPEL Service Engine if monitoring is enabled.
- When implemented with the Database BC, the BPEL Service Engine generates a NullPointerException after shutting down the Database BC.

BPEL Monitoring Console

The following is a known issue for the BPEL Monitoring Console:

- SVG Javascript update does not work on Internet Explorer.
- Performing a search returns the entire result set, which can cause performance issues.
- The bpelconsole web application needs to be deployed on the same server as the BPEL Service Engine.

COBOL Copybook Encoder

The following is a known issue for the COBOL Copybook Encoder:

- NetBeans Issue 178337 (`http://netbeans.org/bugzilla/show_bug.cgi?id=178337`)

  When you run an encoding text on a generated encoder definition XSD file, the Test Encoding window does not always close after the test is processed. The output file is generated to the correct location, so this does not affect the processing of the test.

  To work around this issue, close the window manually.

Data Mashup Service Engine

The following are known issues to the Data Mashup Service Engine:

- Open-DM_DS Issue 163 (`https://open-dm-ds.dev.java.net/issues/show_bug.cgi?id=163`)

  When you create a join when using an Oracle data source and try to display the data in one of the tables, you can only view one page of output. If you click on the next button to view additional data, none is displayed. This is a design-time issue only, and the Data Mashup project works correctly when built and deployed as part of a composite application.
- Issue 158 (`https://open-dm-ds.dev.java.net/issues/show_bug.cgi?id=158`)

When you delete a table from an Enterprise Data Mashup file (by right-clicking the table and selecting Remove Table), the table appears to be removed; but when you save and close the file and then reopen it, the table is still there.

To work around this issue, do the following:

1. Right-click the table to remove and select Remove Table.
2. Save and close the file.
3. Reopen the file.
4. Right-click the table to remove again and select Remove Table.
5. Save and close the file.

   Now when you reopen the file, the table has been removed.

Database Binding Component

The following are known issues for the Database Binding Component:

- Issue 431 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=431`)

  The `ojdbc5.jar` driver for Oracle does not support prepared statements. The following error message is given:

  ```
  java.sql.SQLException: Unsupported feature
  ```

  No resultset is populated, and `java.lang.string` is selected for every parameter, regardless of the datatype.

  To work around this issue, use DataDirect drivers instead of `ojdbc5.jar`.
- Issue 453 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=453`)

  No error message displayed if too many parameters are supplied in an `insert` prepared statement. For example, `insert into table values(?,?,?,?,?)` into a 3-column table causes a runtime exception, and no parameters are added.
- Issue 499 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=499`)

  When a prepared statement is used to update or delete parameters, the result displays the default `java.lang.String` datatype irrespective of the actual datatype. This is because the DataDirect connect of JDBC driver for Oracle does not support returning parameter metadata information for an update statement with mixed literal and parameterized arguments.

  To work around this issue, manually change the datatype of the parameter in the generated XSD.
- Issue 517 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=517`)

  Built-in functions require an alias to avoid misinterpreting names of elements with special characters. For example, the prepared statement `select avg(float) from table where number in(...);` yields the following error:

```
ERROR: s4s-att-invalid-value: Invalid attribute value for 'name' in element
'element'. Recorded reason: cvc-datatype-valid.1.2.1: 'AVG(NUMBER)' is not a
valid value for 'NCName'.
```

To work around this issue, use an alias for the function. For example: `select avg(float)` *avg* `from table where number in(...);`

- Issue 533 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=533`)

  DataDirect driver does not return correct metadata, resulting in an incorrect resultset for output parameters of stored procedures. This is a driver error, and work ticket W809210001 has been filed with DataDirect about the metadata call `getProcedureColumns()` returning incorrect parameters.
- Issue 866 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=866`)

  A datetime value cannot be inserted for SQL Server 2005 using the `sqljdbc.jar` driver. This was caused by a syntax error involving spaces and parentheses.

  See the resolution at `http://forums.microsoft.com/technet/ShowPost.aspx?siteid=17[amp ]amp;postid=4061403`.
- Issue 1087 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=1087`)

  Stored procedures do not support the BLOB (binary large object) datatype. Although the procedure runs without generating an error, a select statement fails to yield any BLOB data.
- Issue 1747 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=1747`)

  When using a Derby database for polling in a clustered environment with the Number of Records option set to a low value, the following exception is thrown and the connection pool becomes unusable:

  ```
  java.sql.SQLException: Connection closed.
  ```

  To work around this issue, configure the number of records per poll to a high number; for example, 100 per 1000 records. A second workaround is to create all cluster-related tables in the same database
- Issue 1803 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=1803`)

  When the Database BC is implemented in a clustered environment across multiple servers, failover does not always work because the message status in the polling table prevents the in-process messages from being picked up by another instance (or by the same instance when it resumes).

  To work around this issue, manually delete records in the polling table that are have a status of "In progress" or "SENT". This allows running instances to pick up the messages and reprocess them.
- When you create a Database BC WSDL document using the wizard, an XSD file named *table_name*`.xsd` is also created. If there is already an existing XSD file by that name, it is overwritten.

  To work around this issue, make sure the table name is different from any existing XSD file names.

Email Binding Component

The following is a known issue for the Email Binding Component:

- The Email BC cannot reconcile email account names that contain a space. For example, if the email account `john.doe@company.com` has an account name of John Doe and the account name is used to populate the From field, the Email BC returns an error when trying to process it.

  To work around this issue, make sure the values populating the From and To fields are actual email addresses that contain no spaces.

File Binding Component

The following are known issues for the File Binding Component:

- The File BC supports encodings in UTF-8 format only; if the encoding is in non-UTF-8 format, the File BC fails to read the input data. However, data of any encoding style (either UTF or non-UTF-8) and binary data can both be forwarded as attachments.

  You can work around the UTF-8 limitation by doing the following:

  - To read non-UTF-8 data, employ a custom encoder upstream of the File BC and convert it to UTF-8 format, which in turn can be provided as input to File BC.
  - To write non-UTF-8 data, route the output from the File BC to a custom encoder and convert it from UTF-8 to the preferred encoding format. For more information, refer to *Designing Custom Encoders*.
- Issue 1206 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=1206`)

  On the Windows operating system, when the File BC transforms XML text to bytes, an additional carriage return is added for each newline character. For example, `\r\n` is converted to `\r\r\n`.

FTP Binding Component

The following is a known issue for the FTP Binding Component:

- Issue 828 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=828`)

  A project with invalid FTP BC property values can be built using command-line builds (ant tasks) but fails validation in NetBeans and is not built. This happens because certain validations that are performed automatically by NetBeans when building a project are not performed by command-line builds (ant tasks). For example, a project with a non-positive or non-integer FTP BC poll value (`pollIntervalMillis`) fails validation in NetBeans but can be built using ant commands.

HL7 Binding Component

The following is a known issue for the HL7 Binding Component:

- Issue 2341 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=2341`)

  The residue memory continues to increase after each sequence of deploy, feed data, undeploy, and forced garbage collection operations related to an HL7 application. Eventually, this will cause the application to run out of memory.

  To work around this issue, restart the GlassFish server when the memory usage become to high.
- For HL7 v2.6 only, the HL7 Binding Component does not support UAC segment validation.

IEP Service Engine

The following are known issues for the IEP Service Engine:

- Issue 449 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=449`)

  When using Oracle as the IEP database platform, you must use Oracle JDBC driver version 10.2.0.4.0 or above. The default driver included with Oracle 9.2 (`ojdbc14.jar`) is version 9.0.2.0.0, and will not work with IEP. To determine the driver version you are using, view the `manifest.mf` file in the driver JAR file.
- OJC Issue 608 (`https://open-jbi-components.dev.java.net/issues/show_bug.cgi?id=608`)

  When a cluster instance is stopped using the asadmin `stop-instance` command, IEP processes that are currently running do not fail over to a live instance. This is due to the order in which methods are called during the stop process.

  To work around this issue, stop and restart the Service Assembly.
- Issue 1519 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=1519`)

  When running the IEP SE on Solaris SPARC using an Oracle database, connection errors might occur.

  To work around this issue, restart the GlassFish server.
- The IEP Service Engine creates connection pools and JDBC resources during installation. This automatic creation takes place only when the IEP Service Engine is installed in a GlassFish Domain Administration Server (DAS) instance. If the IEP Service Engine is installed in a GlassFish standalone instance, then the IEP Service Engine does not create the connection pools and JDBC resources.

  To work around this issue, manually create the connection pools and JDBC resources and then install the IEP Service Engine.

- An IEP Module project can have multiple event processors, and one database connection is created and kept open for each event processor. Therefore, each event processor has a dedicated database connection. When the event processors stop running (for example, when the Composite Application is stopped), this connection is released. Because the IEP SE uses database connections for other tasks, the maximum pool size of the connection pools should be larger than the number of event processors that are running in the IEP SE by a count of 10.
- MySQL does not support microseconds. The IEP Service Engine internally timestamps each incoming event. As a result, the IEP Service Engine cannot differentiate between events that come in less than one second apart. If an incoming message contains any time or timestamp fields, the fractal part of the seconds is ignored from the values of such fields. For example, if the value of a timestamp is '2010–12–10 14:12:09.019473', the fractal portion (019473) is ignored.
- By default, the IEP SE generates a WSDL document for each IEP process document, and regenerates the WSDL document every time the IEP process is edited. By default, bindings and services are generated in this WSDL, and these elements usually need to be edited to work properly when deployed. However, if these WSDL documents are edited, the edits are replaced by the default values every time the WSDL document is regenerated.

  Bindings and services created in the CASA Editor are not affected when the IEP WSDL documents are regenerated. However, you should not clone the IEP WSDL file to customize its generated bindings and services in the CASA Editor because updates to the generated WSDL files are not updated after cloning. Over time, the cloned and edited WSDL deployed for the bindings becomes inconsistent with the WSDL document deployed for the IEP Service Engine. Consider using the following procedure for setting up your Composite Application and IEP project for iterative development:

  1. Define the IEP Module project.
  2. Turn off the generation of concrete configuration elements (bindings and services) in the IEP generated WSDL files by setting the `always.generate.abstract.wsdl` flag in the `project.properties` file to true.
  3. Build the IEP Module project.
  4. Add the IEP Module project to a new Composite Application project and build the project.
  5. Define the binding components and connections using the CASA Editor. To use the composite application test feature, define SOAP input and file output bindings as appropriate for your test environment.
  6. Build and deploy the composite application.
  7. Run tests.
  8. Change the IEP Module project, adjust binding component and service configurations if necessary in the CASA Editor, and rebuild and redeploy the composite application. Run tests and repeat.

Java EE Service Engine

The following is a known issue for the Java EE Service Engine:

- Java EE endpoints are disabled after stopping the service engine, which is expected; but they are not enabled when the service engine is restarted.

  To work around this issue, stop, shut down, and then restart the Java EE Service Engine.

Java Application Project for JAXB

The following is a known issue for Java Application projects with JAXB:

- NetBeans Issue 163169 (`http://www.netbeans.org/issues/show_bug.cgi?id=163169`)

  In Java Application projects for JAXB, JAXB class files are deleted at runtime and the application fails to run.

  To work around this issue, deselect Compile on Save in the Project Properties window. Compile on Save is located in the Build > Compiling properties node.

LDAP Binding Component

The following are known issues for the LDAP Binding Component:

- Issue 804 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=804`)

  When the authentication type is not agreed upon between the LDAP server and client, an `AuthenticationNotSupportedException`occurs. For example, the error occurs when a client specifies strong SSL authentication with a DSEE 6 server that handles none or simple authentication, but not strong authentication.

- Issue 924 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=924`)

  No design-time validation is performed when an invalid root DN is supplied (such as `dc=sun`instead of `dc=sun,dc=com`), and the user is (incorrectly) allowed to create LDAP WSDL files with the incorrect values. Such a project correctly fails at runtime with a `No such object` error message.

PIX/PDQ Manager

The following is a known issue for the PIX/PDQ Manager:

- When you run the automated build scripts for the PIX/PDQ Manager projects, the script might fail because it is unable to delete all pre-existing build files.

  To work around this issue, rerun the script until it completes successfully.

REST Binding Component

The following is a known issue for the REST Binding Component:

- When installing the REST BC in a clustered environment with more than one instance, the same REST BC default HTTP/HTTPS listening ports are assigned for all instances. The port numbers should be unique for each instance.

Worklist Manager Service Engine

The following are known issues for the Worklist Manager Service Engine:

- The default Worklist Manager Console does not support Access Control Lists (ACL) security. This means that users can view and reassign tasks that are not assigned to them.

  To work around this issue, you can customize the console to enforce ACL security.
- For security and authentication purposes, the default Worklist Manager Console must be deployed on the same server that the Worklist Manager Service Engine is installed on.

  To work around this issue, you can customize the console to have a user's identity explicitly resolved by the GlassFish security realm from another server. This requires customizing `index.jsp` in the sample WLM Console project using the methods in `TaskListPage`.

Sun Master Index

The following is a known issue for Sun Master Index:

- Master Index applications created in Java CAPS 6.x will not work in GlassFish ESB.

XSLT Service Engine

The following are known issues for the XSLT Service Engine:

- Issue 1631 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=1631`)

  A SEVERE error appears in the server log when running an XSLT project with an HTTP/SOAP binding. The error is:

  ```
  Unable to create envelope from given source because the root element is not
  named Envelope
  ```

  This is an erroneous log message and does not interfere with the running of the XSLT project.

  To prevent the message from being logged, set the SAAJ Logger for the XSLT SE to **`OFF`**.
- NetBeans Issue 161321 (`http://www.netbeans.org/issues/show_bug.cgi?id=161321`)

If you change text in the XML editor, clicking Validate does not save the changes before compiling and validating the code. You need to click Save All before validating the XML.

Cluster Configuration

The following is a known issue for configuring GlassFish ESB cluster instances:

- Issue 1909 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=1909`)

  When you configure cluster instances using the GlassFish Admin Console, the configuration does not propagate to any new cluster instances. This is because instances are configured individually. In some cases, this can cause deployment to fail.

  To work around this, configure each cluster instance individually or use the following asadmin command to configure all instances in the cluster:

  ```
  asadmin update-jbi-application-variable --component=sun-jms-binding
  name=value
  ```

  Usage: `asadmin update-jbi-application-variable [--terse=false] [--echo=false] [--interactive=true] [--host localhost] [--port 4848|4849] [--secure | -s] [--user admin_user] [--passwordfile file_name] [--target target(Default server)] --component Component Name (name=value)[,name=value]*|property file`
- Installing the GlassFish ESB Platinum Pack in a clustered environment fails due to an issue with the encoder shared library.

  To work around this issue, before you install the Platinum Pack, do the following:

  1. Shut down all components in the cluster that use the encoder shared library, such as the File BC and FTP BC.
  2. Uninstall the encoder library from the cluster.
  3. Stop the cluster and the application server.

     The Platinum Pack should now install successfully in the clustered environment.

Composite Application

The following are know issues for Composite Application (CA) projects:

- NetBeans Issue 152449 (`http://www.netbeans.org/issues/show_bug.cgi?id=152449`)

  When a new project type is freshly installed, the Composite Application project is initially unable to determine the service engine type and is thus unable to deploy the service unit. The following error message appears:

  ```
  Cannot add JBI module project_type.
  Its target component type engine_name is unknown.
  ```

  To work around this issue, stop and then restart the NetBeans IDE.

- NetBeans Issue 153537 (`http://www.netbeans.org/issues/show_bug.cgi?id=153537`)

  You cannot rebuild a Composite Application project after a Service Unit (SU) has been deleted from it.

  To work around this issue, manually edit the `build-impl.xml` file and delete all sections that operate on the deleted SU.

Installation and Upgrading

The following are known issues for installation and upgrading:

- Issue 1981 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=1981`)

  When you perform a side-by-side upgrade of GlassFish ESB, the upgrade tool (`asupgrade`) does not migrate application variables and application configuration.

  To work around this issue, perform the upgrade and then recreate the application variables and application configurations in the new domain for GlassFish ESB 2.1.
- When using GlassFish ESB with JDK 1.6.0_13 or 1.6.0_14, several of the wizards used to develop application will not work correctly and may have problems importing XSDs from files. In addition, the GlassFish ESB Installer fails to install on Windows when using JDK 1.6.0_13 or 1.6.0_14.

  To work around this issue, install a different version of JDK 1.6.
- The Healthcare Pack Installer requires JDK 1.6.0_04 or later, excluding JDK 1.6.0_13 and 1.6.0_14.
- The Platinum Pack silent installation does not allow you to select which components to install. It only performs a complete installation of the Platinum Pack components.
- You cannot change the master password when you perform an installation. The default master password is **`changeit`**. You can change the master password if necessary using the following command:

  `asadmin change-master-password` *new-password*

  Substitute the actual password for *new-password*.
- Before running the component installer against existing NetBeans and GlassFish environments on Mac OS, you need to modify the `netbeans.conf` file by uncommenting and updating the value of the `netbeans_jdkhome` property. This should point to the JDK home directory. For example, `netbeans_jdkhome="/Library/Java/Home"`
- When you run the component installer against an existing GlassFish environment, you cannot specify which domain to install the components to. If there is more than one domain, the components are installed to all domains.
- The component installer does not currently support installing components to a remote domain.
- Installing the Healthcare Pack to a clustered environment fails.

To work around this issue, remove the encoder shared library from the cluster before running the Healthcare Pack Installer.

- If you are upgrading to GlassFish ESB 2.2 and are using BPEL monitoring, you need to manually add two tables to the monitoring database. Here are the SQL Commands to use:

  For MySQL Databases:

  ```
  CREATE TABLE MONITORBPELPROCESS (
      suname varchar (256),
      bpelid varchar (744),
      genbpelevents char(1),
      primary key (bpelid)
  ) ENGINE=InnoDB;
  ```

  For Oracle and Derby Databases:

  ```
  CREATE TABLE MONITORBPELPROCESS (
      suname varchar(256),
      bpelid varchar(1028),
      genbpelevents char(1),
      primary key (bpelid)
  );
  ```

JBI Management

The following are known issues in managing JBI components:

- NetBeans Issue 163169 (`http://www.netbeans.org/issues/show_bug.cgi?id-163169`)

  Selecting the Compile on Save option in a Java Application Project Properties window causes a JAXB class file to be deleted at runtime. The project fails at runtime with the following exception:

  ```
  Exception in thread "main" java.lang.NoClassDefFoundError
  ```

  To work around this issue, deselect the Compile on Save option. This option is located on the Java Application Properties window under Build > Compiling.

Localization

The following is a known localization issue:

- Issue 1138 (`https://open-esb.dev.java.net/issues/show_bug.cgi?id=1138`)

  When you perform the following steps in a non-English locale, you cannot expand the GlassFish server tree to select the value.

  1. Launch the New File Wizard to create a new JCA message-driven bean.
  2. Complete steps 1 through 3 on the wizard.

3. For the Connection URL in step 4, click the ellipses button and start the GlassFish server to obtain the JNDI value.

To work around this issue, use the GlassFish Admin Console to retrieve the JNDI value to enter for the Connection URL. The value is located under Resources > Connectors > Admin Object Resources.