Artix 1.3.1 Release Notes

In this document

This document contains the following sections:

Upgrading from Previous Versions

When upgrading from a previous version of Artix you need to do the following:

- acquire and install new licences if you are upgrading from Artix 1.1. If you have not received your new license please contact your IONA representative.
- recreate any existing Artix 1.1 projects as the Artix 1.3 project format and the Artix 1.1 project format are not compatible.
- disassociate the `.iap` file extension from the Artix Designer if you selected this option when installing Artix 1.1 on a Windows platform.
- regenerate all code generated from WSDL.
- recompile all applications that contain generated code or reference generated header files.

- due to changes in the way Artix generates code for complex types containing elements with `minOccurs=0` or `maxOccurs>1` you will need to change any Artix application code that explicitly uses `ElementListT`. The code will need to be modified as shown by the following example:

```
// Existing code:
IT_Bus::ElementListT<SomeType, &SomeComplexType_x_qname, 0, 10>&
   list = something.getx();
// Correct code:
IT_Vector<SomeType>& list = something.getx()
```

- Due to a change in the Artix servant implementation user code can no longer get the `Port` object directly from within the generated server Impl object. You must now use the `Current` object to get the `Port` object in Artix server code. The code for getting a port using the `Current` object is shown below:

```
void TestImpl::do_stuff() IT_THROW_DECL((IT_Bus::Exception))
  {
    Current& current = get_bus()->get_current();
    Port& port =  current.get_operation().get_port();
  }
```

New Features

The following new features have been added for the Artix 1.3.1 release:

- JMS Transport
- Contract Generation From Java Classes
- Java Client Support
- to_string()
- Security
- User Defined Exceptions in Java
- Type Support
- PIC S9 Support in Fixed Binding

JMS Transport

Artix now supports connections to JMS providers for both C++ and Java applications. The JMS transport can be used to send messages using any of the data bindings supported by Artix and can package them using either a `TextMessage` body or an `ObjectMessage` body.

Note: Currently the Artix JMS transport is only supported for use with the SonicMQ JMS.

Contract Generation From Java Classes

Artix 1.3 includes a tool, `javatowsdl`, that generates Artix contracts from compiled Java class files.

Java Client Support

Artix 1.3 supports the development of Artix clients in Java. Artix Java clients are developed using the dynamic proxy as described in the JAX-RPC 1.1 specification.

to_string()

All Artix generated types now provide a method, `to_string()`, that creates an XML string containing the data as follows:

```
// C++
SomeType x;

// This will give an element called MyElm containing the value of x
IT_Bus::String x_xml = IT_Bus::to_string(x, QName("MyElm"));

// This will give an element with the default name to_string
IT_Bus::String x_xml = IT_Bus::to_string(x);
```

You can also print to an `ostream` using the usual << operator:

```
cout << x << endl;
```

The << operator always uses the default element name `to_string`.

Security

Artix 1.3 has the following new security features:

- Kerberos support.
- WS Security compliance. All security tokens used by Artix comply to the WS Security specification.
- WSDL <port> level security configuration.

User Defined Exceptions in Java

The Artix Java APIs now support user defined exceptions following the JAX-RPC specification. The only differences between Artix generated Java exceptions and truely compliant JAX-RPC Java code is that Artix generates default contructor, setter, and getter methods for each generated exceptions. These default methods are required by the Artix client side runtime.

Type Support

Artix 1.3.1 has extended the Artix type support in the following ways:

- the Artix router supports `AnyType`.
- `AnyType` now supports a value of `Empty`.
- Artix References can now be stored using `AnyType`.

PIC S9 Support in Fixed Binding

The fixed binding supports `PIC S9` and correctly palces the sign bit at the end of the string for fields using this format.

Documentation Updates

The following changes have been made to the Artix documentation for Artix 1.3:

- Title Changes
- User's Guide
- Deploying and Managing Artix Solutions
- Designing Artix Solutions
- Developing Artix Applications in Java
- Developing Artix Applications in C++

The Artix 1.3 documentation is updated regularly on the IONA Support Web Site. Please check there regularly for the latest documentation.

Title Changes

The *Artix C++ Programmer's Guide* has been renamed *Developing Artix Applications in C++*.

User's Guide

The *Artix User's Guide* has been deprecated. The content from the user's guide has been separated into two new books: *Deploying and Managing Artix Applications* and *Designing Artix Applications*.

Deploying and Managing Artix Solutions

This is a new book and contains information needed to configure and deploy Artix solutions. It describes the tasks and information needed to deploy and use the Artix standalone service, the Artix locator, the Artix session manager, and any custom developed Artix applications.

Designing Artix Solutions

This is a new book. It contains the information needed to describe services as Artix contracts. It describes WSDL, all of the WSDL extensions that make up an Artix contract, and how they are used to describe an Artix solution. It also covers how to generate contracts using both Artix Designer and the Artix command line tools.

Developing Artix Applications in Java

This is a new book that addresses how to create Artix applications using the new Artix APIs for Java.

Developing Artix Applications in C++

This guide has been updated to include information on using the Artix session manager.

Known Problems

The following are known problems in this release:

- Demos
- Artix Designer

- Tuxedo Plugin
- Java Threading
- AIX Java Runtime
- HTTP Server-Side Keep-Alive Support
- Security Server

Demos

The following are known issues with the demos:

- To run the `hello_world/soap_mq` demo you need to modify your configuration to add the `mq` plugin into the `orb_plugins` list.
- On HP-UX, running the `secure_hello_world` demo requires the following preperation:

```
$ cd <installation dir>/shlib
$ cp libcrypto.sl libcrypto.1
$ rm libcrypto.sl
$ ln -s libcrypto.1 libcrypto.sl
```

- To run the demo, `switch\run_outside`, you have to put the `mq` plugin before the `routing` plugin in the `orb_plugins` list as follows:

```
iona_services
   {
artix_service
   {
   orb_plugins = ["xmlfile_log_stream", "iiop_profile", "giop",
   "iiop", "soap", "http", "ws_orb", "mq","routing"];
...
}
```

Artix Designer

The following are known issues when using Artix Designer:

- The deployment package generated by Artix Designer cannot be unzipped on Windows XP systems. However, the files in the generated deployment package are also deployed into the specified directory structure by Artix Designer.
- There is currently no support for creating a JMS endpoint.

- There are problems with running the UI with JDK 1.4.2. This problem manifests itself in the form of a stack trace when the user clicks on the development panel under one node, then on another development node immediately after.

Tuxedo Plugin

Tuxedo does not allow you to have a space in the pathname of any path referenced in the `tux.env` script. You need to use the short file name format for Windows environments. You can display the short filenames using `dir /x`.

Java Threading

`THREAD-LOCAL` should not be used when developing Artix Java applications. They are not maintained by the Artix runtime.

AIX Java Runtime

Due to a possible conflict between the version of `xerces` and `xalan` in the AIX JDK and the version that is shipped with Artix you may need to add the following flags when running Artix Java applications:

- `$JDK_ENDORSED_DIRS`
- `$JDK_BOOTSTRAP_CLASSPATH`

HTTP Server-Side Keep-Alive Support

When an Artix HTTP server accepts an incoming connection it immediately reads off the first request to determine the URL/servant. From that point on the connection is then tied to that URL/servant. This is wrong because you may more than one servant on the same http port and they should be accessible by a single connection to that port.

There are a couple of workarounds you can use:

1. Don't use keep-alives. The simplest way to do this is to use the `honorKeepAlive` attribute on the `<http-conf:server>` wsdl extension element to ensure that it never allows keep alives.
2. Ensure the endpoints are running on different tcp ports. You can do this by using fixed ports (or port ranges) or put your servants in different Buses.

Security Server

During the startup, the security server prints out the following messages on HP-UX:

```
/usr/lib/dld.sl: Unresolved symbol: oop_iterate__7oopDescFP10OopClosure (code) from
   /install_dir/jre/lib/PA_RISC/server/libjvm.sl
/usr/lib/dld.sl: Unresolved symbol: _adjust_pointer__9MarkSweepSFPP7oopDescb (code) from
   /install_dir/jre/lib/PA_RISC/server/libjvm.sl
/usr/lib/dld.sl: Unresolved symbol: do_oop_nv__16FilteringClosureFPP7oopDesc (code) from
   /hp/install_dir/jre/lib/PA_RISC/server/libjvm.sl
/usr/lib/dld.sl: Unresolved symbol: oop_iterate__7oopDescFP10OopClosure (code) from
   /install_dir/jre/lib/PA_RISC/server/libjvm.sl
/usr/lib/dld.sl: Unresolved symbol: _adjust_pointer__9MarkSweepSFPP7oopDescb (code) from
   /install_dir/jre/lib/PA_RISC/server/libjvm.sl
/usr/lib/dld.sl: Unresolved symbol: do_oop_nv__16FilteringClosureFPP7oopDesc (code) from
   /install_dir/jre/lib/PA_RISC/server/libjvm.sl
```

The messages occur due to dependancy on the classic JVM, which is deprecated in JDK1.4.X, and can be safely ignored.

Fixed Bugs in Artix 1.3

The following bugs have been closed in Artix 1.3:

Bug #	Description
68490	Artix core dumps `ref` attribute refers to the name of an element defined in another schema.
68461	Artix1.2.2 generated two schema types for each bounded CORBA type in an idl.
68450	`wsdltocpp` does not generate the correct code for recursive wsdl types.
68477	The Artix 1.2.1 Session Manager does not behave as expected when initial location is not `localhost`.
68469	Unable to specify `localhost` in port of wsdl for a session managed server.
68470	Keep-alive does not work for session-managed artix client.

Bug #	Description
68460	Artix Tuxedo server crashes during `tmshutdown`.
68457	Generate code for multiple contracts that each import a common xsd.
68459	Problem with occurrence constraints in return types causing client to throw exception on marshalling.
68282	Artix clients can not handle the `FaultException` thrown by a Weblogic server.
68458	Unable to specify a port range for session managed servers.
68473	The port number for the `artix_wsdl_publishing` plugin needs to be user configurable.
68385	Artix Designer fails to parse WSDL with more than one level of import.
68425	Artix Designer does not remove the CORBA type mapping when it removes the associated CORBA binding.
68238	Artix server should publish wsdl at the `soap:address` location.
68211	Support for operator <<.
68262	Artix clients and servers need an option to dump the http configuration at start up.
68230	A default port should be supplied for `simple_client_server` demo.
68381	Artix does not support xsd `<include .../>` or `<import .../>`.
68353	`$IT_CONFIG_DOMAINS_DIR` and `artix.cfg` missing from Artix runtime install.
68368	Console mode installation on Solaris hangs.
68512	Unable to filter out `pingRequest`/`pingResponse` for session-managed servers.
68481	When you try launch a server on a port that is already active, the server hangs.

Bug #	Description
68497	Artix generates incorrect code for `minoccurs=0, maxoccurs=1`.

Fixed Bugs in Artix 1.3.1

The following bugs have been closed in Artix 1.3.1:

Bug#	Description
271230	Artix 1.3 for Java runtime does not support User defined exception, if a server throws a user defined exception, the client gets a `javax.rmi.RuntimeException`, instead of the UD exception type.
271224	Artix 1.3 client fails when a null is provided for a value even with `minOccurs=0` in the WSDL.
271192	Artix 1.3 runtime does not support Document Literal SOAP encoding.
271168	Artix Designer code generator (Java) generates incorrect syntax for user defined exception types. The first character for the name of the private members was capitalized in the constructor, versus lower case in the declaration.
68656	Cluttered up menus, sub menus and command buttons on the WSDL edit tab.
68655	Confirmation dialogbox when the Designer is closed and the project is not saved.
68653	Server coredumps due to uncaught exception.
68651	The session-managed server coredumps.
68637	Artix xsd attribute does not support special characters like carriage return.
68626	Artix Designer outputs a wrong path to the WSDL in the generated Java server and demo code.

Bug#	Description
68625	`artix_env` does not append/prepend the PATH variable with `$JAVA_HOME/bin`
68623	Artix Designer code generator does not map Java package names to directories before generating the code.
68548	fixed binding : `minOccurs`/`maxOccurs` attributes does not work for elements of basic type.
68529	Located session server core-dumps on startup if `event_log:filter` is set to `"*=*"`.

Reporting Problems

Contact customer support at http://www.iona.com/support/contact/

Other Resources

- IONA University (`http://www.iona.com/info/services/ps/`) delivers practical and insightful courses that cover technical and product issues as well as standards-based best practices gleaned from real-world projects.
- IONA Professional Services (`http://www.iona.com/info/services/global/`) provide product expertise and consulting solutions that empower end-users, system integrators and software vendors with the knowledge to fully leverage IONA products. Together, IONA consultants and products equip you with a single platform for integrating and developing extremely reliable, scalable and secure e-Business systems.
- The latest updates to the Artix documentation can be found at `http://www.iona.com/docs`.
- Knowledge base articles (`http://www2.iona.com/MinervaRoot/index.jsp`): A database that contains practical advice on specific development issues, contributed by IONA developers, support specialists, and customers.