IronJacamar 1.1 Developer's Guide

Connecting your Enterprise Information Systems

IronJacamar 1.1 Developer's Guide: Connecting your Enterprise Information Systems

To all Java EE Connector Architecture users, and especially the IronJacamar community

Preface

1. About IronJacamar

The goal of the IronJacamar project is to provide an implementation of the Java Connector Architecture 1.7 specification.

The specification can be found here: http://www.jcp.org/en/jsr/detail?id=322.

The IronJacamar project is licensed under the GNU LESSER GENERAL PUBLIC LICENSE 2.1 (LGPL 2.1) license.

2. Why IronJacamar ?

The Java EE Connector Architecture container can be viewed as a foundation inside an application server as it provides connectivity to the other containers such that they can communicate with EISes. Iron is often used as foundation in building houses too.

The Jacamar bird family which lives in Central and South America are glossy elegant birds with long bills and tails. Why we picked the Jacamar family is left as an exercise for the reader :)

3. Versions

This section contains the highlights of the IronJacamar releases. A full description of each release can be found through our issue tracking system at http://issues.jboss.org/browse/JBJCA.

3.1. IronJacamar 1.1

Highlights as compared to IronJacamar 1.0:

- Java EE Connector Architecture 1.7 certified (standalone / Java EE7)
- Lazy connection manager (JCA chapter 7.16)
- Distributed work manager (JCA chapter 10.3.11)
- Advanced pool capacity policies and flush strategies
- Enhanced Arquillian integration
- Eclipse development plugin
- Enterprise Information System testing server
- Resource adapter information tool
- Migration tools

3.2. IronJacamar 1.0

Highlights as compared to previous Java EE Connector Architecture containers inside JBoss Application Server:

- Java EE Connector Architecture 1.6 certified (standalone / Java EE6)
- POJO container environment
- New configuration schemas which focuses on usability
- Fast XML and annotation parsing for quick deployment
- Reauthentication support
- Prefill support for security backed domains
- Support for pool flushing strategies
- Embedded environment for ease of development with Arquillian and ShrinkWrap integration
- New management and statistics integration for components
- Code generator for resource adapters
- Validator tool for resource adapters

4. The team

Jesper Pedersen acts as the lead for the IronJacamar project. He can be reached at jesper (dot) pedersen (at) ironjacamar (dot) org.

Jeff Zhang is a core developer on the IronJacamar project. He can be reached at jeff (dot) zhang (at) ironjacamar (dot) org.

Stefano Maestri is a core developer on the IronJacamar project. He can be reached at stefano (dot) maestri (at) ironjacamar (dot) org.

Lin Gao is a core developer on the IronJacamar project. He can be reached at lin (dot) gao (at) ironjacamar (dot) org.

Vladimir Rastseluev is a core developer on the IronJacamar project. He can be reached at vrastseluev (at) ironjacamar (dot) org.

Dimitris Andreadis is an advocate for the IronJacamar project. He can be reached at dimitris (at) ironjacamar (dot) org.

Johnaton Lee helps out in the IronJacamar community with identifying issues, and fixing them. He can be reached at johnathonlee (at) ironjacamar (dot) org.

Tyronne Wickramarathne helps out in the IronJacamar community with identifying issues, and fixing them. He can be reached at tyronne (at) ironjacamar (dot) org.

5. Thanks to

Adrian Brock, Carlo de Wolf, Gurkan Erdogdu, Bruno Georges, Paul Gier, Jason Greene, Stefan Guilhen, Jonathan Halliday, Søren Hilmer, Ales Justin, Vicky Kak, Aslak Knutsen, Sacha Labourey, Mark Little, Alexey Loubyansky, Patrick MacDonald, Scott Marlow, Shelly McGowan, Andrig Miller, Marcus Moyses, Weston Price, Andrew Lee Rubinger, Heiko Rupp, Anil Saldhana, Scott Stark, Clebert Suconic, Andy Taylor, Vladimir Vasilev, Jeremy Whiting, Yang Yong and Leslie York.

6. License

1

Introduction

The Java Connector Architecture (JCA) defines a standard architecture for connecting the Java EE platform to heterogeneous Enterprise Information Systems (EIS). Examples of EISs include Enterprise Resource Planning (ERP), mainframe transaction processing (TP), databases and messaging systems.

The connector architecture defines a set of scalable, secure, and transactional mechanisms that enable the integration of EISs with application servers and enterprise applications.

The connector architecture also defines a Common Client Interface (CCI) for EIS access. The CCI defines a client API for interacting with heterogeneous EISs.

The connector architecture enables an EIS vendor to provide a standard resource adapter for its EIS. A resource adapter is a system-level software driver that is used by a Java application to connect to an EIS. The resource adapter plugs into an application server and provides connectivity between the EIS, the application server, and the enterprise application. The resource adapter serves as a protocol adapter that allows any arbitrary EIS communication protocol to be used for connectivity. An application server vendor extends its system once to support the connector architecture and is then assured of seamless connectivity to multiple EISs. Likewise, an EIS vendor provides one standard resource adapter which has the capability to plug in to any application server that supports the connector architecture.

1.1. What's New

1.1.1. Java Connector Architecture 1.7

The Java Connector Architecture 1.7 specification adds the following areas:

- Adds an activation name for message endpoints to uniquely identify them
- Deployment annotations for connection factories and administration objects

> **Note**
>
> The deployment annotations are only meant for developer usage, and should not be used in test or production environments.

The IronJacamar standalone and embedded distributions doesn't support these annotations.

1.1.2. Java Connector Architecture 1.6

The Java Connector Architecture 1.6 specification adds the following major areas:

- Ease of Development: The use of annotations reduces or completely eliminates the need to deal with a deployment descriptor in many cases. The use of annotations also reduces the need to keep the deployment descriptor synchronized with changes to source code.
- Generic work context contract: A generic contract that enables a resource adapter to control the execution context of a Work instance that it has submitted to the application server for execution.
- Security work context: A standard contract that enables a resource adapter to establish security information while submitting a Work instance for execution to a WorkManager and while delivering messages to message endpoints residing in the application server.
- Standalone Container Environment: A defined set of services that makes up a standalone execution environment for resource adapters.

1.2. Overview

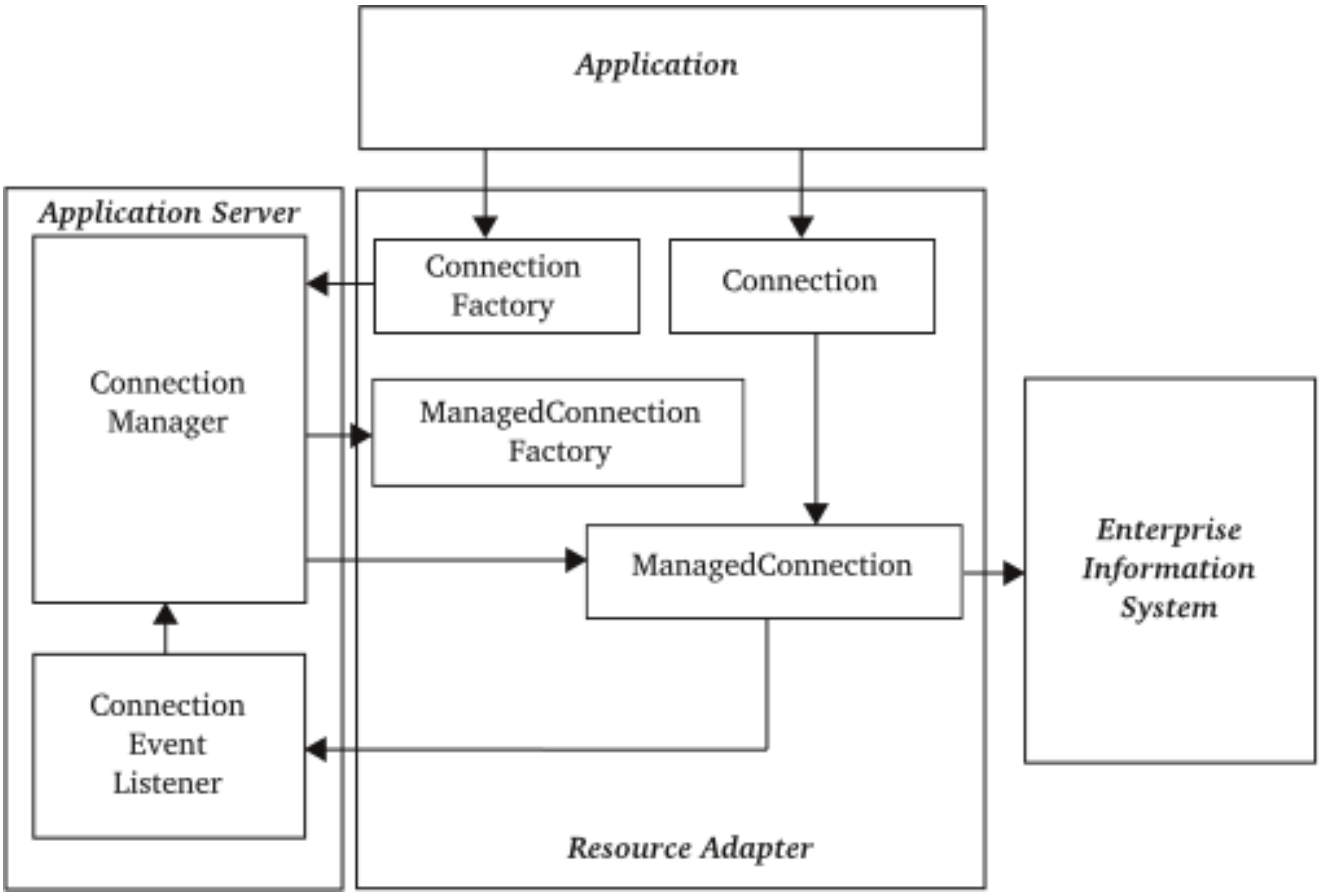
The Java EE Connector Architecture features three different types of resource adapters

- Outbound: The resource adapter allows the application to communicate to the Enterprise Information System (EIS).
- Inbound: The resource adapter allows messages to flow from the Enterprise Information System (EIS) to the application.
- Bi-directional: The resource adapter features both an outbound and an inbound part.

For more information about Java EE Connector Architecture see the specification.

1.2.1. Outbound resource adapter

The Java Connector Architecture specification consists of a number of outbound components:



The application uses the

- ConnectionFactory: The connection factory is looked up in Java Naming and Directory Interface (JNDI) and is used to create a connection.
- Connection: The connection contains the Enterprise Information System (EIS) specific operations.

The resource adapter contains

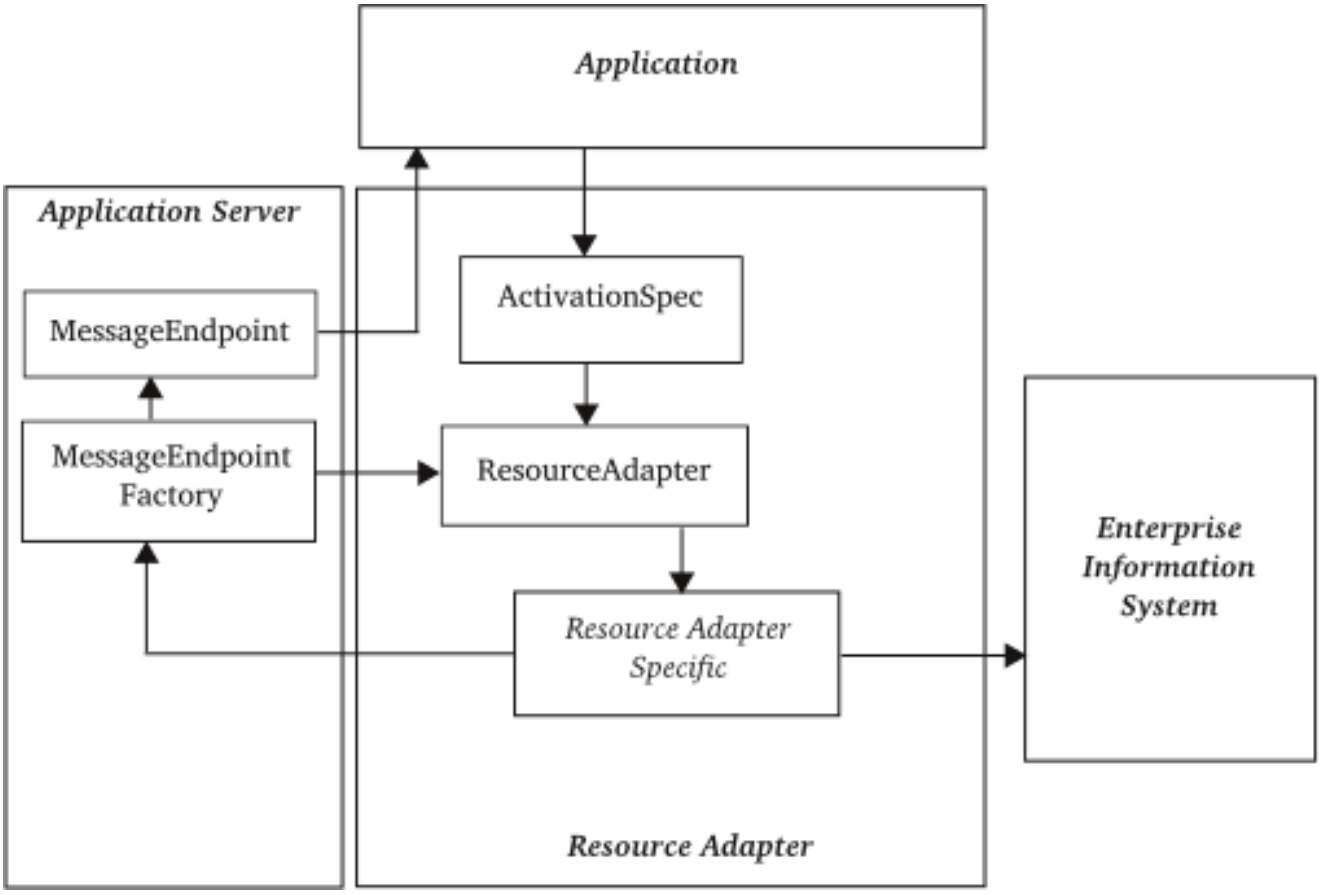
- ManagedConnectionFactory: The managed connection factory creates managed connections.
- ManagedConnection: The managed connection represents a physical connection to the target Enterprise Information System (EIS). The managed connection notifies the application server of events such as connection closed and connection error.

IronJacamar - the application server - contains

- ConnectionManager: The connection manager handles all managed connections in regards to pooling, transaction and security.
- ConnectionEventListener: The connection event listener allows the connection manager to know the status of each managed connection.

1.2.2. Inbound resource adapter

The Java Connector Architecture specification consists of a number of inbound components:



The application uses the

- ActivationSpec: The activation specification specifies the different properties that the application is looking for from the resource adapter and hence the Enterprise Information System (EIS). This specification can be hidden from the user by a facade provided by the application server.

The resource adapter contains

- ResourceAdapter: The resource adapter provides the activation point for inbound communication.
- Resource adapter specific: The resource adapter specific code handles communication with the Enterprise Information System (EIS) and deliver messages through the MessageEndpointFactory.

IronJacamar - the application server - contains

- MessageEndpointFactory: The MessageEndpointFactory is registered with the ResourceAdapter instance and creates the MessageEndpoint instances.

- MessageEndpoint: The MessageEndpoint contains the actual message from the Enterprise Information System (EIS) which the application uses. This could for example be a message driven Enterprise JavaBean (EJB/MDB).

2

Building

2.1. Prerequisites

2.1.1. Java Development Kit (JDK)

You must have the following JDK installed in order to build the project:

- Sun JDK 1.7.x
- OpenJDK 1.7.x

Remember to ensure that "javac" and "java" are in your path (or symlinked).

```
JAVA_HOME=/location/to/javahome
export JAVA_HOME

PATH=$JAVA_HOME/bin:$PATH
export PATH
```

2.1.2. Apache Ant

You must have Apache Ant 1.8.2+ installed on your system.

Remember to ensure that "ant" are in your path (or symlinked).

```
ANT_HOME=/location/to/anthome
export ANT_HOME

PATH=$ANT_HOME/bin:$PATH
export PATH
```

You may need to set the memory settings for the Apache Ant process like

```
```

```
ANT_OPTS="$ANT_OPTS -Xms128m -Xmx512m -XX:MaxPermSize=256m"
export ANT_OPTS
```

2.1.3. Apache Ivy

The IronJacamar project uses Apache Ivy for dependency management.

Apache Ivy is automatically downloaded and included in the development environment, so no additional setup is required.

2.1.4. Git

You must have Git installed on your system.

Remember to ensure that "git" are in your path (or symlinked).

2.2. Obtaining the source code

2.2.1. Forking the repository

The IronJacamar repository is located at:

```
https://github.com/ironjacamar/ironjacamar
```

Press the "Fork" button in order to fork the repository to your own GitHub account.

Clone your repository to your machine using

```
git clone git@github.com:<your_account>/ironjacamar.git
```

Next add the upstream repository as a remote location:

```
cd ironjacamar
git remote add upstream git@github.com:ironjacamar/ironjacamar.git
```

2.2.2. Git branches

We have the following branches for the project:

- master

  The head of development targeting the next upcoming release.

- 1.0

  The development targeting the IronJacamar 1.0 releases.

- 1.1

  The development targeting the IronJacamar 1.1 releases.

2.3. Compiling the source code

In order to build the IronJacamar project you execute:

```
ant <target>
```

where target is one of

- jars

  Builds the JAR archives in the distribution.

- test

  Builds the JAR archives in the distribution and runs all the test cases.

- module-test

  Builds the JAR archives in the distribution and runs all the test cases for the specified module (`-Dmodule=<modulename>`).

- one-test

  Builds the JAR archives in the distribution and runs the specified test case (`-Dmodule=<modulename> -Dtest=<classname>`).

- docs

  Builds the API documentation for the project.

- aggregated-javadocs

  Builds the Aggregated API documentation for the project.

- aggregated-javadocs-api-spi

  Builds the Aggregated API documentation for API/SPI of the project.

- sjc

  Builds the standalone environment using IronJacamar/SJC.

- release

  Builds a release of the project.

- clean

  Cleans the project of temporary files.

- clean-cache

  Cleans the Apache Ivy repository.

See the full list of targets in the main build.xml file.

An example to get the IronJacamar/SJC built and running:

```
ant clean sjc
cd target/sjc/bin
./run.sh
```

2.4. Creating a patch

Our user guide explains in the "I would like to implement a feature" section how to get started on a writing a new feature or submitting a patch to the project.

You should develop your feature on a Git branch using

```
git checkout -b <feature_name>
```

Once you are done you will need to rebase your work with the latest master

```
git fetch upstream
git rebase -i upstream/master
```

You will need to resolve any conflicts of course. Note, that all pull requests must be rebased against upstream master in order to get merged.

Then push the feature to your repository

```
git push origin <feature_name>
```

Go to your account on GitHub.com and submit a pull request via the "Pull request" button

```
https://www.github.com/<your_account>/ironjacamar
```

Remember to select the correct branch, fill in the subject with a short description of the feature, and fill in the description with the full description of the feature.

If your feature / bug-fix applies to multiple branches you will need to submit multiple pull requests - one pull request per branch.

3

Releases

The chapter describes the various releases and their exit criteria.

3.1. Overview

Each release is labelled with a version number and an identifier.

```
ironjacamar-<major>.<minor>.<patch>[.<identifier>]
```

where

- Major: The major version number. Signifies major changes in the implementation.
- Minor: The minor version number. Signifies functional changes to a major version.
- Patch: The patch version number. Signifies a binary compatible change to a minor version.
- Identifier: The identifier. Identifies the level of the quality of the release.
  - None / Final: Stable release
  - CR: Candidate for Release quality. The implementation is functional complete.
  - Beta: Beta quality. The implementation is almost functional complete.
  - Alpha: Alpha quality. The implementation is a snapshot of the development.

3.2. Versioning

Each release will contain a version number which relates to the feature branch where it was created.

3.2.1. Major

A Major version identifier signifies major changes in the implementation such as a change in the architecture.

The features between major versions can be a lot different, and therefore feature regressions may appear.

A Major version will most likely also mean updates to the configuration and required metadata files for deployments.

3.2.2. Minor

A Minor version identifier signifies functional changes to a Major release.

This means that new features have been added to the Major release, and hence may have new configuration options and integration points.

The release is binary compatible to the previous releases - for example `1.0` vs. `1.1`.

3.2.3. Patch

A Patch version identifier signifies a binary compatible update to one or more components in a Minor release.

This means that one or more bug fixes to existing components have been integrated in the branch in question.

The release is binary compatible to the previous releases - for example `1.0.0` vs. `1.0.1`.

3.3. Identifiers

Each release will contain an identifier which relates to the release quality.

3.3.1. Alpha releases

An Alpha release is a snapshot of the main development branch which likely will contain new features.

> **Warning**
> Alpha releases are NOT production quality

An Alpha release are made each month (time-boxed) unless the branch is using an identifier as Beta or higher.

The exit criteria for an Alpha release is that the main test suite is passing.

3.3.2. Beta releases

A Beta release contains major features that are considered almost functional complete. This doesn't mean however that all aspects of each feature is complete and therefore not all options will be active.

> **Warning**
> Beta releases are NOT production quality

A Beta release will be made once one or more features are almost functional complete and therefore Beta releases aren't time-boxed, but feature-boxed instead.

The exit criteria for a Beta release is that all test suites are passing.

3.3.3. Candidate for Release releases

A Candidate for Release is considered functional complete and candidate for being promoted to a Final release.

> **Warning**
> Candidate for Release releases are NOT production quality

A Candidate for Release focuses on functionality, but they are time-boxed to a maximum of two weeks between each release.

The exit criteria for a Candidate for Release release is that all test suites are passing.

3.3.4. Final releases

A Final release is considered feature complete and stable.

Typically only one Final release will be released from each branch, unless critical or blocker issues are found in the release. Patch releases will be available from our source control system as tags.

The exit criteria for a Final release is that all test suites are passing.

3.4. Nexus

The IronJacamar artifacts are uploaded to the JBoss.org Nexus repository located at:

```
https://repository.jboss.org/nexus/content/groups/public/
```

The IronJacamar artifacts are deployed under the `groupId` of:

```
org.jboss.ironjacamar
```

See the User Guide for a complete list of artifacts.

3.4.1. Deploying a release

A release of IronJacamar is deployed using:

```
ant nexus
cd target
./deploy.sh
```

After the artifacts have been uploaded the release must be promoted in Nexus by logging in and choosing "Promote" and "Close". The path for the staging repository can be used for testing the release.

Note, that this requires Maven 3.0.5+.

3.4.2. Deploying a snapshot

A snapshot of IronJacamar is deployed using:

```
ant -Dsnapshot=true nexus
cd target
./deploy.sh
```

Note, that this requires Maven 3.0.5+.

3.4.3. Deploying a snapshot (locally)

A snapshot of IronJacamar is deployed to the local Maven repository `$HOME/.m2/repository` using:

```
ant -Dsnapshot=true nexus
cd target
./install.sh
```

Note, that this requires Maven 3.0.5+.

4

Issue tracking

4.1. Location

The JIRA issue tracking for the project is located at http://issues.jboss.org/browse/JBJCA
.

4.2. Components

The project is divided into the following components:

Table 4.1. Project components

Component	Description
Arquillian	The Arquillian integration for the project.
AS	The tools that focuses on integration with WildFly.
Build	The build environment for the project.
Code Generator	The resource adapter code generator.
Common	Common interfaces and classes that are shared between multiple components.
Core	The core implementation of the project.
Documentation	The documentation (Users Guide / Developers Guide) for the project.
Deployer	The deployers for the project.
Eclipse	The Eclipse plugin for IronJacamar.
EIS	The EIS test server.
Embedded	The embedded IronJacamar container.
JDBC	A JDBC resource adapter.
Performance	Performance related work.
Standalone	The standalone IronJacamar distribution.
Test Suite	The IronJacamar test suite.
Validator	The resource adapter validator.

4.3. Categories

The system contains the following categoies:

Table 4.2. JIRA categories

Category	Description
Feature Request	Request for a feature made by the community.
Bug	Software defect in the project.
Task	Development task created by a member of the team.
Release	Issue which holds informations about a release.
Component Update	Identifies a thirdparty library dependency.

The other categories in the JIRA installation are not used by this project.

4.4. Life cycle

All issues folows the following life cycle:

Table 4.3. JIRA Lifecycle

Lifecycle	Description
Open	An issue currently not implemented.
Coding in Progress	An issue currently being worked on.
Resolved	An issue which has been implemented.
Closed	An issue that has been resolved and is included in a release.

Note: 'Component Update' issues can't be resolved nor closed during a development cycle. These are resolved and closed as part of the release procedure of the project. The reason for this is that the library in question can receive further updates during the active development cycle.

4.5. Priorities

All issues are assigned one of the following priorities:

Table 4.4. JIRA Priorities

Priority	Description
Blocker	An issue that needs to be fixed before the release.
Critical	An issue that is critical for the release.
Major	The default priority for an issue.

Priority	Description
Minor	An issue that is optional for a release.
Trivial	An issue that is optional for a release and have a lower priority than Minor.

5

Testing

5.1. Overall goals

The overall goals of our test environment is to execute tests that ensures that we have full coverage of the JCA specification as well as our implementation.

The full test suite is executed using

```
ant test
```

A single test case can be executed using

```
ant -Dmodule=embedded -Dtest=org.jboss.jca.embedded.unit.ShrinkWrapTestCase one-test
```

where `-Dmodule` specifies which module to execute the test case in. This parameter defaults to `core`. The `-Dtest` parameter specifies the test case itself.

You can also execute all test cases of a single module using

```
ant -Dmodule=embedded module-test
```

where `-Dmodule` specifies which module to execute the test cases in. This parameter defaults to `core`.

The build script does not fail in case of test errors or failure.

You can control the behavior by using the `junit.haltonerror` and `junit.haltonfailure` properties in the main `build.xml` file. Default value for both is `no`.

You can of course change them statically in the `build.xml` file or temporary using `-Djunit.haltonerror=yes`. There are other `jnuit.*` properties defined in the main `build.xml` that can be controlled in the same way.

5.1.1. Specification

The purpose of the specification tests is to test our implementation against the actual specification text.

Each test can only depend on:

- The official Java Connector Architecture API (javax.resource)
- Interfaces and classes in the test suite that extends/implements the official API

The test cases should be created in such a way such that they are easily identified by chapter, section and paragraph. For example:

```
org.jboss.jca.core.spec.chaper10.section3
```

5.1.2. IronJacamar specific interfaces

The purpose of the IronJacamar specific interfaces tests is to test our specific interfaces.

Each test can depend on:

- The official Java Connector Architecture API (javax.resource)
- The IronJacamar specific APIs (org.jboss.jca.xxx.api)
- Interfaces and classes in the test suite that extends/implements these APIs

The test cases lives in a package that have a meaningful name of the component it tests. For example:

```
org.jboss.jca.core.workmanager
```

These test cases can use both the embedded JCA environment or be implemented as standard POJO based JUnit test cases.

5.1.3. IronJacamar specific implementation

The purpose of the IronJacamar specific implementation tests is to test our specific implementation. These tests should cover all methods are not exposed through the interface.

Each test can depend on:

- The official Java Connector Architecture API (javax.resource)
- The IronJacamar specific APIs (org.jboss.jca.xxx.api)
- The IronJacamar specific implementation (org.jboss.jca.xxx.yyy)
- Interfaces and classes in the test suite

The test cases lives in a package that have a meaningful name of the component it tests. For example:

```
org.jboss.jca.core.workmanager
```

These test cases can use both the embedded JCA environment or be implemented as standard POJO based JUnit test cases.

5.2. Testing principle and style

Our tests follows the Behavior Driven Development (BDD) technique. In BDD you focus on specifying the behaviors of a class and write code (tests) that verify that behavior.

You may be thinking that BDD sounds awfully similar to Test Driven Development (TDD). In some ways they are similar: they both encourage writing the tests first and to provide full coverage of the code. However, TDD doesn't really provide a guide on which kind of tests you should be writing.

BDD provides you with guidance on how to do testing by focusing on what the behavior of a class is supposed to be. We introduce BDD to our testing environment by extending the standard JUnit 4.x test framework with BDD capabilities using assertion and mocking frameworks.

The BDD tests should

- Clearly define `given-when-then` conditions
- The method name defines what is expected: f.ex. shouldReturnFalseIfMethodXIsCalledWithNullString()
- Easy to read the assertions by using Hamcrest Matchers [http://code.google.com/p/hamcrest/]
- Use `given` facts whenever possible to make the test case more readable. It could be the name of the deployed resource adapter, or using the BDD Mockito class [http://mockito.googlecode.com/svn/branches/1.8.0/javadoc/org/mockito/BDDMockito.html] to mock the fact.

We are using two different kind of tests:

- Integration Tests: The goal of these test cases is to validate the whole process of deployment, and interacting with a sub-system by simulating a critical condition.
- Unit Tests: The goal of these test cases is to stress test some internal behaviour by mocking classes to perfectly reproduce conditions to test.

5.2.1. Integration Tests

The integration tests simulate a real condition using a particular deployment artifacts packaged as resource adapters.

The resource adapters are created using either the main build environment or by using ShrinkWrap [http://community.jboss.org/wiki/ShrinkWrap]. Using resource adapters within the test cases will allow you to debug both the resource adapters themself or the JCA container.

The resource adapters represent the [given] facts of our BDD tests, the deployment of the resource adapters represent the [when] phase, while the [then] phase is verified by assertion.

Note that some tests consider an exception a normal output condition using the JUnit 4.x `@Exception(expected = "SomeClass.class")` annotation to identify and verify this situation.

5.2.2. Unit Tests

We are mocking our input/output conditions in our unit tests using the Mockito [http://mockito.googlecode.com] framework to verify class and method behaviors.

An example:

```
@Test
public void printFailuresLogShouldReturnNotEmptyStringForWarning() throws Throwable
{
   //given
   RADeployer deployer = new RADeployer();
   File mockedDirectory = mock(File.class);
   given(mockedDirectory.exists()).willReturn(false);

   Failure failure = mock(Failure.class);
   given(failure.getSeverity()).willReturn(Severity.WARNING);

   List failures = Arrays.asList(failure);
   FailureHelper fh = mock(FailureHelper.class);
   given(fh.asText((ResourceBundle) anyObject())).willReturn("myText");

   deployer.setArchiveValidationFailOnWarn(true);

   //when
   String returnValue = deployer.printFailuresLog(null, mock(Validator.class),
                                                  failures, mockedDirectory, fh);

   //then
   assertThat(returnValue, is("myText"));
}
```

As you can see the BDD style respects the test method name and using the `given-when-then` sequence in order.

5.3. Quality Assurance

In addition to the test suite the IronJacamar project deploys various tools to increase the stability of the project.

The following sections will describe each of these tools.

5.3.1. Checkstyle

Checkstyle is a tool that verifies that the formatting of the source code in the project is consistent.

This allows for easier readability and a consistent feel of the project.

The goal is to have zero errors in the report. The checkstyle report is generated using

```
ant checkstyle
```

The report is generated into

```
reports/checkstyle
```

The home of checkstyle is located here: http://checkstyle.sourceforge.net/.

5.3.2. Findbugs

Findbugs is a tool that scans your project for bugs and provides reports based on its findings.

This tool helps lower of the number of bugs found in the IronJacamar project.

The goal is to have zero errors in the report and as few exclusions in the filter as possible. The findbugs report is generated using

```
ant findbugs
```

The report is generated into

```
reports/findbugs
```

The home of findbugs is located here: http://findbugs.sourceforge.net/.

5.3.3. JaCoCo

JaCoCo generates a test suite matrix for your project which helps you identify where you need additional test coverage.

The reports that the tool provides makes sure that the IronJacamar project has the correct test coverage.

The goal is to have as high code coverage as possible in all areas. The JaCoco report is generated using

```
ant jacoco
```

The report is generated into

```
reports/jacoco
```

The home of JaCoCo is located here: http://www.eclemma.org/jacoco/.

5.3.4. Tattletale

Tattletale generates reports about different quality matrix of the dependencies within the project.

The reports that the tool provides makes sure that the IronJacamar project doesn't for example have cyclic dependencies within the project.

The goal is to have as no issues flagged by the tool. The Tattletale reports are generated using

```
ant tattletale
```

The reports are generated into

```
reports/tattletale
```

The home of Tattletale is located here: http://www.jboss.org/tattletale.

5.4. Performance testing

Performance testing can identify areas that needs to be improved or completely replaced.

5.4.1. JProfiler

Insert the following line in `run.sh` or `run.bat`:

```
-agentpath:<path>/jprofiler6/bin/linux-x64/libjprofilerti.so=port=8849
```

where the Java command is executed.

The home of JProfiler is located here: http://www.ej-technologies.com/products/jprofiler/overview.html.

5.4.2. OProfile

OProfile can give a detailed overview of applications running on the machine, including Java program running with OpenJDK.

The home of OProfile is located here: http://oprofile.sourceforge.net.

5.4.2.1. Installation

Enable the Fedora debug repo:

```
/etc/yum.repos.d/fedora.repo

[fedora-debuginfo]
name=Fedora $releasever - $basearch - Debug
failovermethod=priority
mirrorlist=https://mirrors.fedoraproject.org/metalink?repo=fedora-debug-$releasever&arch=
$basearch
enabled=1
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-fedora-$basearch
```

Install:

```
yum install -y oprofile oprofile-jit
yum install -y yum-plugin-auto-update-debug-info
yum install -y java-1.6.0-openjdk-debuginfo
```

```
```

5.4.2.2. Running

Insert the following line in `run.sh` or `run.bat`:

```
-agentpath:/usr/lib64/oprofile/libjvmti_oprofile.so
```

for 64bit JVMs or

```
-agentpath:/usr/lib/oprofile/libjvmti_oprofile.so
```

for 32 bit JVMs where the Java command is executed.

Now execute:

```
opcontrol --no-vmlinux
opcontrol --start-daemon
```

and use the following commands:

```
opcontrol --start # Starts profiling
opcontrol --dump  # Dumps the profiling data out to the default file
opcontrol --stop  # Stops profiling
```

Once you are done execute:

```
opcontrol --shutdown  # Shuts the daemon down
```

A report can be generated by:

```
```

```
opreport -l --output-file=<filename>
```

Remember that this is system wide profiling, so make sure that only the services that you want included are running.

More information is available at http://oprofile.sourceforge.net/doc/index.html.

6

Metadata

6.1. Core Metadata

The metadata for the IronJacamar project is split up into the following areas

- Java Connector Architecture Metadata
- IronJacamar Metadata
- Resource adapter deployment Metadata
- DataSource deployment Metadata

All metadata parsing is done using the StAX model (`javax.xml.stream`) for optimal performance.

The implementation of these areas is done within the common module of the project.

6.1.1. Java Connector Architecture Metadata

The Java Connector Architecture (JCA) metadata implement the metadata defined in the JCA specifications. We have metadata representing the following standards:

- Java Connector Architecture 1.0
- Java Connector Architecture 1.5
- Java Connector Architecture 1.6

An instance of the metadata is bundle with the resource adapter archive as

```
META-INF/ra.xml
```

The implementation is split into two package hierarchies - the API in

```
org.jboss.jca.common.api.metadata.ra
```

and the implementation in

```
org.jboss.jca.common.metadata.ra
```

.

6.1.2. IronJacamar Metadata

The IronJacamar metadata can provide overrides for the values specified in the standard Java Connector Architecture metdata. It is also possible to specify deployment metadata, which will active an instance of the resource adapter without any other deployment plans.

Supported versions of the metadata:

- IronJacamar 1.0

An instance of the metadata is bundle with the resource adapter archive as

```
META-INF/ironjacamar.xml
```

The implementation is split into two package hierarchies - the API in

```
org.jboss.jca.common.api.metadata.ironjacamar
```

and the implementation in

```
org.jboss.jca.common.metadata.ironjacamar
```

.

6.1.3. Resource adapter deployment Metadata

The resource adapter deployment metadata provides a deployment plan for the specified resource adapter archive. It is possible to override metadata specified as part of the Java Connector Architecture metadata or the IronJacamar metadata.

Supported versions of the metadata:

- Resource adapter deployment 1.0

The implementation is split into two package hierarchies - the API in

```
org.jboss.jca.common.api.metadata.resourceadapters
```

and the implementation in

```
org.jboss.jca.common.metadata.resourceadapters
```

.

6.1.4. Datasource deployment Metadata

The datasource deployment metadata provides a deployment plan for datasources. The metadata allows the developer to setup connection parameters, pooling settings and security.

Supported versions of the metadata:

- Datasource deployment 1.0

The implementation is split into two package hierarchies - the API in

```
org.jboss.jca.common.api.metadata.ds
```

and the implementation in

```
org.jboss.jca.common.metadata.ds
```

.

6.1.4.1. Datasource mapping

The table below specifies how each attribute/element map to the resource adapter or the container.

Table 6.1. Datasource mapping

Tag	Resource Adapter	Container
min-pool-size		Pool
max-pool-size		Pool
prefill		Pool
user-name	X	
password	X	
connection-url	X	
driver-class	X	
transaction-isolation	X	
connection-property	X	
url-delimiter	X	
url-selector-strategy-class-name	X	
new-connection-sql	X	
xa-datasource-property	X	
xa-datasource-class	X	
is-same-rm-override		TxConnectionManager
interleaving		TxConnectionManager
prepared-statement-cache-size	X	
share-prepared-statements	X	
pad-xid		TxConnectionManager
wrap-xa-resource		TxConnectionManager
no-tx-separate-pools		Pool
jndi-name		ConnectionManager
pool-name		X
enabled		X
use-java-context		X
valid-connection-checker-class-name	X	
check-valid-connection-sql	X	
validate-on-match	X	

Tag	Resource Adapter	Container
background-validation		Pool
background-validation-minutes		Pool
use-fast-fail		Pool
stale-connection-checker-class-name	X	
exception-sorter-class-name	X	
blocking-timeout-millis		Pool
idle-timeout-minutes		Pool
set-tx-query-timeout		
query-timeout	X	
use-try-lock	X	
allocation-retry		ConnectionManager
allocation-retry-wait-millis		ConnectionManager
xa-resource-timeout		TxConnectionManager
track-statements	X	
prepared-statement-cache-size	X	
share-prepared-statements	X	

6.2. Metadata Repository

The metadata repository serves as a central point for all the metadata in the systems.

6.2.1. Interface

The interface of the metadata repository is located in:

```
org.jboss.jca.core.spi.MetaDataRepository
```

providing methods to query and update the repository.

6.2.2. Bean

The implementation of the metadata repository can be defined as:

```
<bean name="MetaDataRepository"
      interface="org.jboss.jca.core.spi.MetaDataRepository"
      class="org.jboss.jca.core.mdr.SimpleMetaDataRepository">
</bean>
```

which is a simple implementation of the metadata repository service provider interface (SPI).

7

Deployers

The deployer chains for the project is located in the `deployers` module.

7.1. RAR Deployer

The responsibility of the RAR deployer is to deploy a resource adapter archive (.RAR) file.

7.1.1. Fungal

The Fungal kernel features a simple deployment framework, so only three classes are needed for the deployer chain.

The classes are located in the

```
deployers/src/main/java/org/jboss/jca/deployers/fungal
```

directory.

7.1.1.1. RADeployer

This class represent a resource adapter deployer and implements the

```
com.github.fungal.spi.deployers.Deployer
com.github.fungal.spi.deployers.MultiStageDeployer
com.github.fungal.spi.deployers.DeployerOrder
```

interfaces.

The responsible of the class is to

- Create a classloader for the deployment
- Retrieve metadata and annotations such that they can be merged

- Perform archive validation using the JCA validator
- Perform bean validation
- Register the metadata in the metadata repository
- Register the resource adapter in the metadata repository
- Identify and activate the resource adapter objects - if JNDI information is available
- Bind connection factories and admin objects into JNDI - if JNDI information is available

If the resource adapter isn't activated in this step based on an `ironjacamar.xml` file, the deployment will advance to the next step in the deployer chain.

7.1.1.2. RADeployment

This class represent a resource adapter deployment and implements the

```
com.github.fungal.spi.deployers.Deployment
```

interface.

The responsible of the class is to

- Unregister the resource adapter from the metadata repository
- Unregister the JNDI bindings in the metadata repository - if the deployment was activated
- Unbind connection factories and admin objects in JNDI - if the deployment was activated
- Close the classloader
- Clean up any temporary files

7.1.1.3. RaXmlDeployer

This class represent a resource adapter deployer and implements the

```
com.github.fungal.spi.deployers.Deployer
com.github.fungal.spi.deployers.MultiStageDeployer
com.github.fungal.spi.deployers.DeployerOrder
com.github.fungal.spi.deployers.DeployerPhases
```

interface.

The class deploys resource adapter archives based on a `-ra.xml` which provides the necessary deployment information.

The responsible of the class is to

- Create a classloader for the deployment
- Retrieve metadata from the metadata repository
- Merge metadata from the deployment descriptor
- Perform archive validation using the JCA validator
- Perform bean validation
- Register the metadata in the metadata repository
- Identify and activate the resource adapter objects
- Bind connection factories and admin objects into JNDI

Since multiple resource adapter archives can be activated within a single `-ra.xml` file the class uses the `DeployerPhases` callbacks to unregister these from the container. If there is only a single resource adapter activation the deployer acts as part of the normal deployer chain.

7.1.1.4. RaXmlDeployment

This class represent a resource adapter deployment from the `RaXmlDeployer` and implements the

```
com.github.fungal.spi.deployers.Deployment
```

interface.

The responsible of the class is to

- Unregister the JNDI bindings in the metadata repository
- Unbind connection factories and admin objects in JNDI
- Close the classloader

7.1.1.5. RAActivator

This class will activate all resource adapters which hasn't been deployed by a previous step. The class implements the

```
```

```
com.github.fungal.spi.deployers.DeployerPhases
```

interface. This interface allows the class to hook into the deployer lifecycle of the kernel and receive callback notifications.

The responsible of the class is to

- Find any resource adapters which hasn't been deployed through the metadata repository
- Perform a deployment like `RADeployer`
- Register each deployment with the kernel through the main deployer

7.1.1.6. RAActivatorDeployment

This class represent a resource adapter deployment activated by the `RAActivator` and implements the

```
com.github.fungal.spi.deployers.Deployment
```

interface.

The responsible of the class is to

- Unregister the JNDI bindings in the metadata repository
- Unbind connection factories and admin objects in JNDI
- Close the classloader

7.2. DataSource Deployer

The responsibility of the datasource deployer is to deploy a datasource deployment (-ds.xml) file.

7.2.1. Fungal

The Fungal datasource deployer chain consists of two classes.

The classes are located in the

```
deployers/src/main/java/org/jboss/jca/deployers/fungal
```

directory.

7.2.1.1. DsXmlDeployer

This class represent a datasource deployer and implements the

```
com.github.fungal.spi.deployers.Deployer
```

interface.

The responsible of the class is to

- Locate metadata about JDBC in the metadata repository
- Activate each `DataSource` using `jdbc-local.rar` as a template
- Activate each `XaDataSource` using `jdbc-xa.rar` as a template

7.2.1.2. DsXmlDeployment

This class represent a datasource deployment and implements the

```
com.github.fungal.spi.deployers.Deployment
```

interface.

The responsible of the class is to

- Unbind the datasource in JNDI
- Close the classloader

8

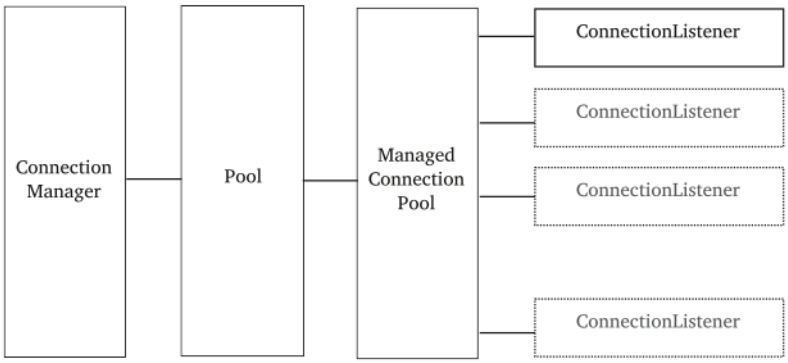
Connection manager

8.1. Overview

The connection manager defines the interface for resource adapters to allocate connections, which are associated with the physical connections to the target Enterprise Information System, such as a database.

It is up to the connection manager to use the pool, and enlist the connection listeners in the transactions, if supported.

The connection manager architecture



Note

Each of the components in the connection manager architecture can be accessed by multiple threads at the same time, hence each component needs to make sure that thread safely is maintained.

8.2. Public API

The public API defines the API that programs external to the IronJacamar project can use to configure, and use the connection manager.

The `ConnectionManager` interface allows to associate and dissociate a managed connection with a connection.

The `ConnectionListener` interface provides with IronJacamar contract for `javax.resource.spi.ConnectionEventListener`.

The package for the public API is `org.jboss.jca.core.api.connectionmanager`.

8.3. Private API

The private API defines the API that can be used internally IronJacamar to control the connection manager. The API extends the public API where it makes sense in order to provide a more uniform interface to the implementation.

The `ConnectionManagerFactory` class can create a `ConnectionManager` instance.

The `ConnectionManager` interface defines the internal API of the connection manager used by IronJacamar.

The `NoTxConnectionManager` interface defines the internal API of a non-transactional connection manager.

The `TxConnectionManager` interface defines the internal API of a transactional connection manager.

The `ConnectionListener` interface defines the internal API of a connection listener, which is associated with a managed connection.

The package for the private API is `org.jboss.jca.core.connectionmanager`.

8.4. Implementation

The implementation of the connection manager is split in two classes, with a shared base class for common functionality.

8.4.1. AbstractConnectionManager

`AbstractConnectionManager` provides the methods that are shared across both implementations, `NoTxConnectionManagerImpl` and `TxConnectionManagerImpl`.

`getManagedConnection(Transaction, Subject, ConnectionRequestInfo)` obtains a `ConnectionListener` instance from the pool. If an error occurs a retry is performed, if configured.

`returnManagedConnection(ConnectionListener, boolean)` returns a `ConnectionListener` instance to the pool. If an error occurs a retry is performed, if configured.

`allocateConnection(ManagedConnectionFactory, ConnectionRequestInfo)` allocates a connection for the managed connection attached to the `ConnectionListener` instance.

`associateManagedConnection(Object, ManagedConnectionFactory, ConnectionRequestInfo)` associates a `ManagedConnection` with the passed in connection. This is a `LazyAssociatableConnectionManager` extension of IronJacamar.

`dissociateManagedConnection(Object, ManagedConnection, ManagedConnectionFactory)` dissociates a connection with its `ManagedConnection` instance. This is a `LazyAssociatableConnectionManager` extension of IronJacamar.

`shutdown` shuts down the connection manager, and its related pool instance.

8.4.2. NoTxConnectionManagerImpl

`NoTxConnectionManagerImpl` provides the implementation for a connection manager configured with `NoTransaction`.

`createConnectionListener(ManagedConnection, ManagedConnectionPool)` creates a `NoTxConnectionListener` and associates it with the `ManagedConnection` instance.

8.4.3. TxConnectionManagerImpl

`TxConnectionManagerImpl` provides the implementation for a connection manager configured with `LocalTransaction` or `XATransaction`.

`createConnectionListener(ManagedConnection, ManagedConnectionPool)` creates a `TxConnectionListener` and associates it with the `ManagedConnection` instance. In case of `LocalTransaction` a `XAResource` is created to wrap the local transaction methods of the resource adapter. In case of `XATransaction` the underlying `XAResource` instance is wrapped with product information, if configured.

`getManagedConnection(Subject, ConnectionRequestInfo)` verifies that a valid transaction exists before creating a `ManagedConnection`, since there is no point in spending resources on work that is going to be rolled back. `ironjacamar.allow_marked_for_rollback` overrides this behavior.

`managedConnectionReconnected(ConnectionListener)` enlist the `ConnectionListener` in the transaction.

`managedConnectionDisconnected(ConnectionListener)` delist the `ConnectionListener` from the transaction.

`lazyEnlist(ManagedConnection)` handles lazy enlistment scenarios as defined by `LazyEnlistableConnectionManager`.

8.4.4. AbstractConnectionListener

`AbstractConnectionListener` provides the methods that are shared across both implementations, `NoTxConnectionListener` and `TxConnectionListener`. This base class keeps track off the connection handles used for the associated `ManagedConnection`.

`connectionErrorOccurred(ConnectionEvent)` logs the application error, and flushes either the `Pool` or `ManagedConnectionPool` depending on the strategy configured.

`controls(ManagedConnection, Object)` checks if the `ManagedConnection` and optional connection is controlled by this connection listener.

`compareTo(Object)` is used to sort the connection listeners based on their last used time.

8.4.5. NoTxConnectionListener

`NoTxConnectionListener` is the listener for `NoTransaction` scenarios.

`connectionClosed(ConnectionEvent)` dissociates the connection handles, and if there are no handles associated anymore the `ManagedConnection` is returned to the pool.

8.4.6. TxConnectionListener

`TxConnectionListener` is the listener for `LocalTransaction` and `XATransaction` scenarios.

`used()` updates the last used time, and resets the timeout value for the underlying `XAResource` if in `XATransaction` mode.

`enlist()` enlists the `XAResource` instances in the transaction through `TransactionSynchronization` including resources picked up by the `CachedConnectionManager`.

`delist()` delists the `XAResource` from the transaction in interleaved scenarios.

`dissociate()` dissociates the `ConnectionListener` with the transaction.

`connectionClosed(ConnectionEvent)` dissociates a connection handle through `wasFreed(Object)` and returns the `ManagedConnection` in interleaved scenarios.

`connectionErrorOccurred(ConnectionEvent)` clears any `TransactionSynchronization` object such that the `ManagedConnection` can be returned for destruction.

`tidyup()` will rollback any left over `LocalTransaction` instance.

`isManagedConnectionFree()` checks if there is exists a `TransactionSynchronization` object in track by transaction scenarios, since the `ManagedConnection` can't be returned in that case.

`wasFreed(Object)` dissociates a connection handle from the `ConnectionListener`, or resets the track by transaction flag if `null` such that the `ManagedConnection` can be returned.

The `TransactionSynchronization` class takes care of enlisting the `XAResource` in the transaction, in track by transaction scenarios. This is done in its `enlist()` and its result can be verified in `checkEnlisted()`. The `beforeCompletion()` method delists the `XAResource` from the transaction. The `afterCompletion(int)` method returns the `ManagedConnection` to the pool.

9

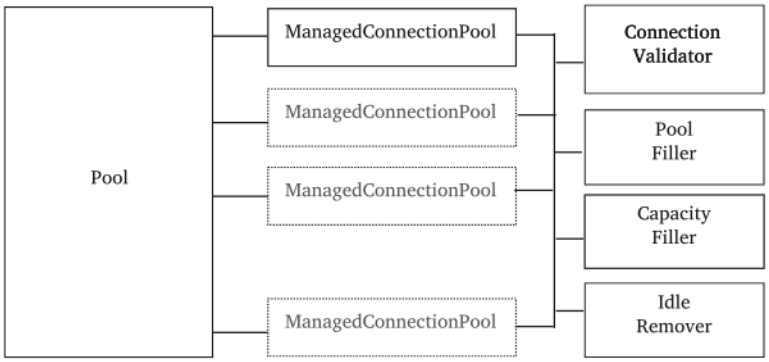
Pool

9.1. Overview

The pool controls the physical connection to the target Enterprise Information System, such as a database.

It is up to the pool to create, hand out and destroy connections in the defined lifecycle based on the configuration parameters supplied by the user.

The pool architecture



Note

Each of the components in the pool architecture can be accessed by multiple threads at the same time, hence each component needs to make sure that thread safely is maintained.

9.2. Public API

The public API defines the API that programs external to the IronJacamar project can use to configure, and use the pool.

The `Pool` interface allows access to the name of the pool, flushing connections, and verifying if a connection can be obtained from the pool.

The `PoolConfiguration` class holds the configuration parameters for the pool.

The `FlushMode` enum defines the different flush modes supported.

The `PoolStatistics` interface defines the statistics available for a pool.

The package for the public API is `org.jboss.jca.core.api.connectionmanager.pool`.

9.3. Private API

The private API defines the API that can be used internally IronJacamar to control the pool. The API extends the public API where it makes sense in order to provide a more uniform interface to the implementation.

The `PoolFactory` class will create a `Pool` based upon the passed in configuration.

The `PoolStrategy` enum defines how the pool is split based on credential equallity.

The `Pool` interface extends the public API with methods that should only be available from with IronJacamar such as getting a connection listener, returning a connection listener, and shutting down the pool.

The `PrefillPool` interface defines the contract for pool implementations that supports prefilling of connections upon startup, such as `OnePool` and `PoolBySubject`.

The `Capacity` interface provides a handle to the policies used for increasing and decreasing the pool.

The `CapacityIncrementer` interface defines if a physical connection should be created given the input parameters.

The `CapacityDecrementer` interface defines if a physical connection should be destroyed given the input parameters.

The package for the private API is `org.jboss.jca.core.connectionmanager.pool.api`.

9.4. Implementation

The pool implementation provides a concrete implementation of the contracts defined by the public and private APIs.

The package for the pool implementation is `org.jboss.jca.core.connectionmanager.pool`.

9.4.1. AbstractPool

`AbstractPool` provides the methods that are shared across all pool implementations.

`getKey(Subject, ConnectionRequestInfo, boolean)` defines the key used to lookup the `ManagedConnectionPool` instance. The implementation of this method is different for each pool type.

`getManagedConnectionPool(Object, Subject, ConnectionRequestInfo)` retrieves the correct `ManagedConnectionPool` instance. If the `ManagedConnectionPool` doesn't yet exists then one is created, and initialized.

`emptyManagedConnectionPool(ManagedConnectionPool)` removes a `ManagedConnectionPool` instance, if unused.

`flush` flushes the `ManagedConnectionPool` instances, based on the `FlushMode`.

`getConnection(Transaction, Subject, ConnectionRequestInfo)` returns a `ConnectionListener` instance, which has the physical connection to the Enterprise Information System attached. The method uses 3 sub methods to return the correct listener instance. `getSimpleConnection` returns a `ConnectionListener` if there is no transaction associated. `getTransactionOldConnection` returns the `ConnectionListener` already associated with the transaction, if any. `getTransactionNewConnection` creates a new `ConnectionListener`, and associates it with the transaction.

`findConnectionListener` finds a specific `ConnectionListener` instance.

`returnConnectionListener` returns a `ConnectionListener` instance to the correct `ManagedConnectionPool`.

`shutdown` shuts down the pool.

`internalTestConnection(ConnectionRequestInfo, Subject)` tries to obtain a `ConnectionListener` based on the input given.

9.4.2. AbstractPrefillPool

The `AbstractPrefillPool` enables the pool to prefill connections during startup, and through its lifecycle.

`prefill(Subject, ConnectionRequestInfo, boolean)` handles the prefilling process.

9.4.3. Pool types

IronJacamar features 5 different pool types. Each pool type has its own `getKey` method implementation that defines how a `ManagedConnectionPool` instance is located.

`OnePool` uses one `ManagedConnectionPool` instance to hold all `ConnectionListeners`.

`PoolByCri` splits the `ManagedConnectionPool` instances based on the `ConnectionRequestInfo` instance.

`PoolBySubject` splits the `ManagedConnectionPool` instances based on the `Subject` instance.

`PoolBySubjectAndCri` splits the `ManagedConnectionPool` instances based on both the `ConnectionRequestInfo` instance and the `Subject` instance.

`ReauthPool` allows the `ConnectionListener` instances to reauthenticate, so the `ManagedConnectionPool` instances can change over time based on the `ConnectionRequestInfo` and `Subject` instances.

The package for the pool types is `org.jboss.jca.core.connectionmanager.pool.strategy`.

9.5. ManagedConnectionPool

The `ManagedConnectionPool` controls the `ConnectionListener` instances, which each has a physical connection (`ManagedConnection`) associated.

The package is `org.jboss.jca.core.connectionmanager.pool.mcp`

9.5.1. Private API

`ManagedConnectionPool` instances should only be accessed from within IronJacamar, so they only have a private API.

The `ManagedConnectionPoolFactory` class creates a `ManagedConnectionPool` instance.

The `ManagedConnectionPool` interface defines the methods exposed to the pool, connection validator, and idle remover. These methods includes getting a connection listener, finding a connection listener, and returning a connection listener.

The `ManagedConnectionPoolStatistics` interface defines the statistics for the `ManagedConnectionPool` instance.

The `ManagedConnectionPoolUtility` class defines utility methods for the `ManagedConnectionPool` instance.

9.5.2. Implementation

There are three different implementations of the `ManagedConnectionPool` interface. `SemaphoreArrayListManagedConnectionPool` which uses a `Semaphore` to guard an `ArrayList` which holds the `ConnectionListeners`. `ArrayBlockingQueueManagedConnectionPool` which uses an `ArrayBlockingQueue` to hold the `ConnectionListeners`. And `LeakDumperManagedConnectionPool` which extends `SemaphoreArrayListManagedConnectionPool`, but reports any leaks upon shutdown.

`getConnection(Subject, ConnectionRequestInfo)` provides a `ConnectionListener`. The method requires a lock in order to obtain a listener, using the specified timeout value. If a listener is avaiable in the pool then it is matched against the `ManagedConnectionFactory` to verify it is valid, and returned - otherwise it is destroyed. If no listener is available then a new listener is created and returned. In the latter case both prefill and a capacity increase is scheduled in order to prefill to the minimum size, and increase the pool by the specified capacity policy.

`returnConnection(ConnectionListener, boolean, boolean)` returns a `ConnectionListener` into the pool.

`flush(FlushMode)` flushes the `ConnectionListeners` according to the mode. Any listeners marked as bad will be destroyed. Prefill is scheduled at the end in order to maintain the minimum pool size.

`removeIdleConnections` is invoked from the idle remover in order to decrement the pool to the desired size based on the `CapacityDecrementer` policy. If any listeners are destroyed the pool is either scheduled for prefill, or for removal through `emptyManagedConnectionPool` if empty.

`shutdown` shuts the instance down. All listeners are removed.

`fillTo(int)` fills the pool to the specified size. The pool filler component uses this method.

`increaseCapacity(Subject, ConnectionRequestInfo)` increases the pool based on the `CapacityIncrementer` policy. The capacity filler component uses this method.

`validateConnections` validates that the listeners are valid according to `ValidatingManagedConnectionFactory`. Any invalid listeners are destroyed and prefill scheduled. The connection validator component uses this method.

`detachConnectionListener` disassociates the connections attached to the `ManagedConnection` such that it can be reused in another request through `DissociatableManagedConnection`.

10

Standalone

10.1. Overview

The standalone IronJacamar container implements Chapter 3 Section 5 of the JCA 1.6 specification which defines a standalone JCA environment.

The standalone container has the following layout:

- `$IRON_JACAMAR_HOME/bin/`

  contains the run scripts and the SJC kernel.

- `$IRON_JACAMAR_HOME/config/`

  contains the configuration of the container.

- `$IRON_JACAMAR_HOME/deploy/`

  contains the user deployments.

- `$IRON_JACAMAR_HOME/doc/`

  contains the documentation.

- `$IRON_JACAMAR_HOME/lib/`

  contains all the libraries used by the container.

- `$IRON_JACAMAR_HOME/log/`

  contains the log files.

- `$IRON_JACAMAR_HOME/system/`

  contains system deployments files.

- `$IRON_JACAMAR_HOME/tmp/`

  contains temporary files.

To start the container execute the following

```
cd $IRON_JACAMAR_HOME/bin
./run.sh
```

.

10.2. IronJacamar/SJC

> **Warning**
> This standalone configuration is for development purposes only.

The IronJacamar/SJC uses the Fungal kernel for its run-time environment.

The homepage for the Fungal is http://jesperpedersen.github.com/fungal
.

SJC is short for "Simple JCA Container".

Appendix A. Licenses

All licenses can be found in the `doc/licenses` directory.

A.1. GNU Lesser General Public License 2.1

GNU LESSER GENERAL PUBLIC LICENSE

Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc. 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

A.1.1. Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method:

1. we copyright the library, and
2. we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the *Lesser* General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a “work based on the library” and a “work that uses the library”.

The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

A.1.2. Terms and Conditions for Copying, Distribution and Modification

A.1.2.1. Section 0

This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called “this License”). Each licensee is addressed as “you”.

A “library” means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The “Library”, below, refers to any such software library or work which has been distributed under these terms. A “work based on the Library” means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term “modification”.)

“Source code” for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

A.1.2.2. Section 1

You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

A.1.2.3. Section 2

You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a. The modified work must itself be a software library.

b. You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c. You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d. If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

   (For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

A.1.2.4. Section 3

You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

A.1.2.5. Section 4

You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

A.1.2.6. Section 5

A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a “work that uses the Library”. Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a “work that uses the Library” with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a “work that uses the library”. The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a “work that uses the Library” uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

A.1.2.7. Section 6

As an exception to the Sections above, you may also combine or link a “work that uses the Library” with the Library to produce a work containing portions of the Library, and distribute that work under

terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a. Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable “work that uses the Library”, as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b. Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c. Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d. If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e. Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the “work that uses the Library” must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

A.1.2.8. Section 7

You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined

library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a. Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b. Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

A.1.2.9. Section 8

You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

A.1.2.10. Section 9

You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

A.1.2.11. Section 10

Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

A.1.2.12. Section 11

If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

A.1.2.13. Section 12

If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

A.1.2.14. Section 13

The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and “any later version”, you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

A.1.2.15. Section 14

If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

A.1.2.16. NO WARRANTY Section 15

BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT

WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.1.2.17. Section 16

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

A.1.3. How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.> Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a “copyright disclaimer” for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library `Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990 Ty Coon, President of Vice

That's all there is to it!

A.2. Creative Commons Attribution–Share Alike 3.0 Unported License

THE WORK (AS DEFINED BELOW) IS PROVIDED UNDER THE TERMS OF THIS CREATIVE COMMONS PUBLIC LICENSE ("CCPL" OR "LICENSE"). THE WORK IS PROTECTED BY COPYRIGHT AND/OR OTHER APPLICABLE LAW. ANY USE OF THE WORK OTHER THAN AS AUTHORIZED UNDER THIS LICENSE OR COPYRIGHT LAW IS PROHIBITED.

BY EXERCISING ANY RIGHTS TO THE WORK PROVIDED HERE, YOU ACCEPT AND AGREE TO BE BOUND BY THE TERMS OF THIS LICENSE. TO THE EXTENT THIS LICENSE MAY BE CONSIDERED TO BE A CONTRACT, THE LICENSOR GRANTS YOU THE RIGHTS CONTAINED HERE IN CONSIDERATION OF YOUR ACCEPTANCE OF SUCH TERMS AND CONDITIONS.

A.2.1. Definitions

a. *"Adaptation"* means a work based upon the Work, or upon the Work and other pre-existing works, such as a translation, adaptation, derivative work, arrangement of music or other alterations of a literary or artistic work, or phonogram or performance and includes cinematographic adaptations or any other form in which the Work may be recast, transformed, or adapted including in any form recognizably derived from the original, except that a work that constitutes a Collection will not be considered an Adaptation for the purpose of this License. For the avoidance of doubt, where the Work is a musical work, performance or phonogram, the synchronization of the Work in timed-relation with a moving image ("synching") will be considered an Adaptation for the purpose of this License.

b. *"Collection"* means a collection of literary or artistic works, such as encyclopedias and anthologies, or performances, phonograms or broadcasts, or other works or subject matter other than works listed in Section 1(f) below, which, by reason of the selection and arrangement of their contents, constitute intellectual creations, in which the Work is included in its entirety in unmodified form along with one or more other contributions, each constituting separate and independent works in themselves, which together are assembled into a collective whole. A work that constitutes a Collection will not be considered an Adaptation (as defined below) for the purposes of this License.

c. *"Creative Commons Compatible License"* means a license that is listed at http://creativecommons.org/compatiblelicenses that has been approved by Creative Commons as being essentially equivalent to this License, including, at a minimum, because that license: (i) contains terms that have the same purpose, meaning and effect as the License Elements of this License; and, (ii) explicitly permits the relicensing of adaptations of works made available under that license under this License or a Creative Commons jurisdiction license with the same License Elements as this License.

d. *"Distribute"* means to make available to the public the original and copies of the Work or Adaptation, as appropriate, through sale or other transfer of ownership.

e. *"License Elements"* means the following high-level license attributes as selected by Licensor and indicated in the title of this License: Attribution, ShareAlike.

f. *"Licensor"* means the individual, individuals, entity or entities that offer(s) the Work under the terms of this License.

g. *"Original Author"* means, in the case of a literary or artistic work, the individual, individuals, entity or entities who created the Work or if no individual or entity can be identified, the publisher; and in addition (i) in the case of a performance the actors, singers, musicians, dancers, and other persons who act, sing, deliver, declaim, play in, interpret or otherwise perform literary or artistic works or expressions of folklore; (ii) in the case of a phonogram the producer being the person or legal entity who first fixes the sounds of a performance or other sounds; and, (iii) in the case of broadcasts, the organization that transmits the broadcast.

h. *"Work"* means the literary and/or artistic work offered under the terms of this License including without limitation any production in the literary, scientific and artistic domain, whatever may be the mode or form of its expression including digital form, such as a book, pamphlet and other writing; a lecture, address, sermon or other work of the same nature; a dramatic or dramatico-musical work; a choreographic work or entertainment in dumb show; a musical composition with or without words; a cinematographic work to which are assimilated works expressed by a process analogous to cinematography; a work of drawing, painting, architecture, sculpture, engraving or lithography; a photographic work to which are assimilated works expressed by a process analogous to photography; a work of applied art; an illustration, map, plan, sketch or three-dimensional work relative to geography, topography, architecture or science; a performance; a broadcast; a phonogram; a compilation of data to the extent it is protected as a copyrightable work; or a work performed by a variety or circus performer to the extent it is not otherwise considered a literary or artistic work.

i. *"You"* means an individual or entity exercising rights under this License who has not previously violated the terms of this License with respect to the Work, or who has received express permission from the Licensor to exercise rights under this License despite a previous violation.

j. *"Publicly Perform"* means to perform public recitations of the Work and to communicate to the public those public recitations, by any means or process, including by wire or wireless means or public digital performances; to make available to the public Works in such a way that members of the public may access these Works from a place and at a place individually chosen by them;

to perform the Work to the public by any means or process and the communication to the public of the performances of the Work, including by public digital performance; to broadcast and rebroadcast the Work by any means including signs, sounds or images.

k. *"Reproduce"* means to make copies of the Work by any means including without limitation by sound or visual recordings and the right of fixation and reproducing fixations of the Work, including storage of a protected performance or phonogram in digital form or other electronic medium.

A.2.2. Fair Dealing Rights

Nothing in this License is intended to reduce, limit, or restrict any uses free from copyright or rights arising from limitations or exceptions that are provided for in connection with the copyright protection under copyright law or other applicable laws.

A.2.3. License Grant

Subject to the terms and conditions of this License, Licensor hereby grants You a worldwide, royalty-free, non-exclusive, perpetual (for the duration of the applicable copyright) license to exercise the rights in the Work as stated below:

a. to Reproduce the Work, to incorporate the Work into one or more Collections, and to Reproduce the Work as incorporated in the Collections;

b. to create and Reproduce Adaptations provided that any such Adaptation, including any translation in any medium, takes reasonable steps to clearly label, demarcate or otherwise identify that changes were made to the original Work. For example, a translation could be marked "The original work was translated from English to Spanish," or a modification could indicate "The original work has been modified.";

c. to Distribute and Publicly Perform the Work including as incorporated in Collections; and,

d. to Distribute and Publicly Perform Adaptations.

e. For the avoidance of doubt:

   i. *Non-waivable Compulsory License Schemes.* In those jurisdictions in which the right to collect royalties through any statutory or compulsory licensing scheme cannot be waived, the Licensor reserves the exclusive right to collect such royalties for any exercise by You of the rights granted under this License;

   ii. *Waivable Compulsory License Schemes.* In those jurisdictions in which the right to collect royalties through any statutory or compulsory licensing scheme can be waived, the Licensor waives the exclusive right to collect such royalties for any exercise by You of the rights granted under this License; and,

   iii. *Voluntary License Schemes.* The Licensor waives the right to collect royalties, whether individually or, in the event that the Licensor is a member of a collecting society that

administers voluntary licensing schemes, via that society, from any exercise by You of the rights granted under this License.

The above rights may be exercised in all media and formats whether now known or hereafter devised. The above rights include the right to make such modifications as are technically necessary to exercise the rights in other media and formats. Subject to Section 8(f), all rights not expressly granted by Licensor are hereby reserved.

A.2.4. Restrictions

The license granted in Section 3 above is expressly made subject to and limited by the following restrictions:

a. You may Distribute or Publicly Perform the Work only under the terms of this License. You must include a copy of, or the Uniform Resource Identifier (URI) for, this License with every copy of the Work You Distribute or Publicly Perform. You may not offer or impose any terms on the Work that restrict the terms of this License or the ability of the recipient of the Work to exercise the rights granted to that recipient under the terms of the License. You may not sublicense the Work. You must keep intact all notices that refer to this License and to the disclaimer of warranties with every copy of the Work You Distribute or Publicly Perform. When You Distribute or Publicly Perform the Work, You may not impose any effective technological measures on the Work that restrict the ability of a recipient of the Work from You to exercise the rights granted to that recipient under the terms of the License. This Section 4(a) applies to the Work as incorporated in a Collection, but this does not require the Collection apart from the Work itself to be made subject to the terms of this License. If You create a Collection, upon notice from any Licensor You must, to the extent practicable, remove from the Collection any credit as required by Section 4(c), as requested. If You create an Adaptation, upon notice from any Licensor You must, to the extent practicable, remove from the Adaptation any credit as required by Section 4(c), as requested.

b. You may Distribute or Publicly Perform an Adaptation only under the terms of: (i) this License; (ii) a later version of this License with the same License Elements as this License; (iii) a Creative Commons jurisdiction license (either this or a later license version) that contains the same License Elements as this License (e.g., Attribution-ShareAlike 3.0 US)); (iv) a Creative Commons Compatible License. If you license the Adaptation under one of the licenses mentioned in (iv), you must comply with the terms of that license. If you license the Adaptation under the terms of any of the licenses mentioned in (i), (ii) or (iii) (the "Applicable License"), you must comply with the terms of the Applicable License generally and the following provisions: (I) You must include a copy of, or the URI for, the Applicable License with every copy of each Adaptation You Distribute or Publicly Perform; (II) You may not offer or impose any terms on the Adaptation that restrict the terms of the Applicable License or the ability of the recipient of the Adaptation to exercise the rights granted to that recipient under the terms of the Applicable License; (III) You must keep intact all notices that refer to the Applicable License and to the disclaimer of warranties with every copy of the Work as included in the Adaptation You Distribute or Publicly Perform; (IV) when You Distribute or Publicly Perform the Adaptation, You

may not impose any effective technological measures on the Adaptation that restrict the ability of a recipient of the Adaptation from You to exercise the rights granted to that recipient under the terms of the Applicable License. This Section 4(b) applies to the Adaptation as incorporated in a Collection, but this does not require the Collection apart from the Adaptation itself to be made subject to the terms of the Applicable License.

c. If You Distribute, or Publicly Perform the Work or any Adaptations or Collections, You must, unless a request has been made pursuant to Section 4(a), keep intact all copyright notices for the Work and provide, reasonable to the medium or means You are utilizing: (i) the name of the Original Author (or pseudonym, if applicable) if supplied, and/or if the Original Author and/or Licensor designate another party or parties (e.g., a sponsor institute, publishing entity, journal) for attribution ("Attribution Parties") in Licensor's copyright notice, terms of service or by other reasonable means, the name of such party or parties; (ii) the title of the Work if supplied; (iii) to the extent reasonably practicable, the URI, if any, that Licensor specifies to be associated with the Work, unless such URI does not refer to the copyright notice or licensing information for the Work; and (iv) , consistent with Section 3(b), in the case of an Adaptation, a credit identifying the use of the Work in the Adaptation (e.g., "French translation of the Work by Original Author," or "Screenplay based on original Work by Original Author"). The credit required by this Section 4(c) may be implemented in any reasonable manner; provided, however, that in the case of a Adaptation or Collection, at a minimum such credit will appear, if a credit for all contributing authors of the Adaptation or Collection appears, then as part of these credits and in a manner at least as prominent as the credits for the other contributing authors. For the avoidance of doubt, You may only use the credit required by this Section for the purpose of attribution in the manner set out above and, by exercising Your rights under this License, You may not implicitly or explicitly assert or imply any connection with, sponsorship or endorsement by the Original Author, Licensor and/or Attribution Parties, as appropriate, of You or Your use of the Work, without the separate, express prior written permission of the Original Author, Licensor and/or Attribution Parties.

d. Except as otherwise agreed in writing by the Licensor or as may be otherwise permitted by applicable law, if You Reproduce, Distribute or Publicly Perform the Work either by itself or as part of any Adaptations or Collections, You must not distort, mutilate, modify or take other derogatory action in relation to the Work which would be prejudicial to the Original Author's honor or reputation. Licensor agrees that in those jurisdictions (e.g. Japan), in which any exercise of the right granted in Section 3(b) of this License (the right to make Adaptations) would be deemed to be a distortion, mutilation, modification or other derogatory action prejudicial to the Original Author's honor and reputation, the Licensor will waive or not assert, as appropriate, this Section, to the fullest extent permitted by the applicable national law, to enable You to reasonably exercise Your right under Section 3(b) of this License (right to make Adaptations) but not otherwise.

A.2.5. Representations, Warranties and Disclaimer

UNLESS OTHERWISE MUTUALLY AGREED TO BY THE PARTIES IN WRITING, LICENSOR OFFERS THE WORK AS-IS AND MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND CONCERNING THE WORK, EXPRESS, IMPLIED, STATUTORY OR OTHERWISE,

INCLUDING, WITHOUT LIMITATION, WARRANTIES OF TITLE, MERCHANTIBILITY, FITNESS FOR A PARTICULAR PURPOSE, NONINFRINGEMENT, OR THE ABSENCE OF LATENT OR OTHER DEFECTS, ACCURACY, OR THE PRESENCE OF ABSENCE OF ERRORS, WHETHER OR NOT DISCOVERABLE. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES, SO SUCH EXCLUSION MAY NOT APPLY TO YOU.

A.2.6. Termination

a. This License and the rights granted hereunder will terminate automatically upon any breach by You of the terms of this License. Individuals or entities who have received Adaptations or Collections from You under this License, however, will not have their licenses terminated provided such individuals or entities remain in full compliance with those licenses. Sections 1, 2, 5, 6, 7, and 8 will survive any termination of this License.

b. Subject to the above terms and conditions, the license granted here is perpetual (for the duration of the applicable copyright in the Work). Notwithstanding the above, Licensor reserves the right to release the Work under different license terms or to stop distributing the Work at any time; provided, however that any such election will not serve to withdraw this License (or any other license that has been, or is required to be, granted under the terms of this License), and this License will continue in full force and effect unless terminated as stated above.

A.2.7. Miscellaneous

a. Each time You Distribute or Publicly Perform the Work or a Collection, the Licensor offers to the recipient a license to the Work on the same terms and conditions as the license granted to You under this License.

b. Each time You Distribute or Publicly Perform an Adaptation, Licensor offers to the recipient a license to the original Work on the same terms and conditions as the license granted to You under this License.

c. If any provision of this License is invalid or unenforceable under applicable law, it shall not affect the validity or enforceability of the remainder of the terms of this License, and without further action by the parties to this agreement, such provision shall be reformed to the minimum extent necessary to make such provision valid and enforceable.

d. No term or provision of this License shall be deemed waived and no breach consented to unless such waiver or consent shall be in writing and signed by the party to be charged with such waiver or consent.

e. This License constitutes the entire agreement between the parties with respect to the Work licensed here. There are no understandings, agreements or representations with respect to the Work not specified here. Licensor shall not be bound by any additional provisions that may appear in any communication from You. This License may not be modified without the mutual written agreement of the Licensor and You.

f. The rights granted under, and the subject matter referenced, in this License were drafted utilizing the terminology of the Berne Convention for the Protection of Literary and Artistic Works (as amended on September 28, 1979), the Rome Convention of 1961, the WIPO Copyright Treaty of 1996, the WIPO Performances and Phonograms Treaty of 1996 and the Universal Copyright Convention (as revised on July 24, 1971). These rights and subject matter take effect in the relevant jurisdiction in which the License terms are sought to be enforced according to the corresponding provisions of the implementation of those treaty provisions in the applicable national law. If the standard suite of rights granted under applicable copyright law includes additional rights not granted under this License, such additional rights are deemed to be included in the License; this License is not intended to restrict the license of any rights under applicable law.

A.3. Apache License, Version 2.0

Apache license

Version 2.0, January 2004

http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

A.3.1. Definitions

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

A.3.2. Grant of Copyright License

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

A.3.3. Grant of Patent License

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

A.3.4. Redistribution

You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

a. You must give any other recipients of the Work or Derivative Works a copy of this License; and

b. You must cause any modified files to carry prominent notices stating that You changed the files; and

c. You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

d. If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

A.3.5. Submission of Contributions

Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

A.3.6. Trademarks

This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

A.3.7. Disclaimer of Warranty

Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

A.3.8. Limitation of Liability

In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

A.3.9. Accepting Warranty or Additional Liability

While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.