

## 10 Tools

### 10.1 Purpose

The Tools section describes the third party tools and open source tools that will be used in development of the Broker along side WebSphere.

### 10.2 Eclipse

Eclipse is an open source community, whose projects are focused on building an extensible development platform, runtimes and application frameworks for building, deploying and managing software across the entire software lifecycle. The UI SIDES use for Eclipse is mainly as a Java IDE, a testing/code review tool, and our interface into the configuration management repository.

The Eclipse open source community has over 60 open source projects. These projects can be conceptually organized into seven different "pillars" or categories:

1. Enterprise Development
2. Embedded and Device Development
3. Rich Client Platform
4. Rich Internet Applications
5. Application Frameworks
6. Application Lifecycle Management (ALM)
7. Service Oriented Architecture (SOA)

The Eclipse community is also supported by a large and vibrant ecosystem of major IT solution providers, innovative start-ups, universities and research institutions and individuals that extend, support and complement the Eclipse Platform.

#### 10.2.1 PMD (Eclipse plug-in)

PMD is a static Java source code analysis tool. It searches Java code for inefficient code, bugs, common coding problems, and other such issues. PMD can be used in the development environment through IDE integrations, or it can be incorporated directly into an Ant or Maven build. PMD uses rules to perform the source code analysis, and the rules are grouped into rule sets.

The rules are categorized by the sort of problem they check for - thus the unused code rule set finds unused local variables and private fields and methods, the strict exception rule set finds methods that throw Exception and catch blocks that catch NullPointerException, and so forth. There are also library-specific rule sets. For example, there's a JUnit rule set that finds common problems (such as using `assert(x==null)` vs `assertNotNull(x)`) in JUnit test suites. Currently we've got around 225 rules and there are more in the pipeline.

PMD can also look for other potential problems like:

- Possible bugs - empty try/catch/finally/switch statements
- Dead code - unused local variables, parameters and private methods
- Suboptimal code - wasteful String/StringBuffer usage
- Overcomplicated expressions - unnecessary if statements, for loops that could be while loops
- Duplicate code - copied/pasted code means copied/pasted bugs

### **10.3 JUNIT**

JUnit is a unit testing framework for the Java programming language. JUnit is one of the xUnit family of frameworks. JUnit has spawned its own ecosystem of JUnit extensions. The following advantages can be gained from using JUnit:

- JUnit tests allow you to write code faster while increasing quality.
- JUnit is elegantly simple.
- JUnit tests check their own results and provide immediate feedback.
- JUnit tests can be composed into a hierarchy of test suites.
- Writing JUnit tests is inexpensive. JUnit is free.
- JUnit tests increase the stability of software.
- JUnit tests are developer tests.
- JUnit tests are written in Java.

An example JUnit test follows:

```
public class HelloWorld extends TestCase
{
    public void testMultiplication()
    {
        // Testing if 3*2=6:
        assertEquals ("Multiplication", 6, 3*2);
    }
}
```

### **10.4 SOAPUI**

soapUI is a free and open source desktop application for

- [inspecting Web Services](#)
- [invoking Web Services](#)
- [developing Web Services](#)
- [Web Services Simulation and Mocking](#)
- [Functional, Load and Compliance testing of Web Services](#)

It is mainly aimed at developers and testers providing or consuming WSDL or REST based Web Services (Java, .net, etc). Functional and Load Testing can be done both

interactively in soapUI or within an automated build or integration process using the soapUI [command line tools](#).

Mock Web Services can easily be created for any WSDL and hosted from within soapUI or using the command-line MockService runner. IDE-plugins are available for

- [eclipse plug in](#)
- [IntelliJ IDEA plug in](#)
- [NetBeans plug in](#)

One of the main uses of SOAP UI on SIDES is for load testing.

The following features for load testing web services are currently available:

Create any number of [Web Service LoadTests](#) for a TestCase

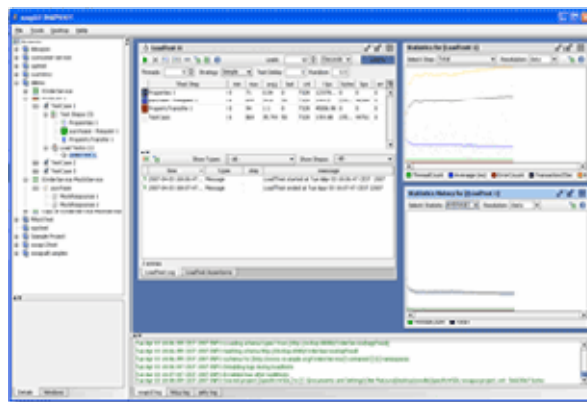
Choose between configurable [Web Service Load Strategies](#), [Limits](#) and [Thread-counts](#) and analyze how web services perform under a variety of scenarios

[Assert LoadTest results](#) continuously for performance and functionality surveillance

[Web Service usage Behavioral Diagrams](#) allow real time analysis of performance statistics

Export results, statistics, logs, diagram data, etc. for external processing

Run multiple LoadTests interactively in soapUI, through Maven or [from a soapUI command line](#)



### **10.5 *BadBoy***

Badboy is a powerful tool designed to aid in testing and development of complex dynamic applications. Badboy makes web testing and development easier with dozens of features including a simple yet comprehensive capture/replay interface, powerful load testing support, detailed reports, and graphs.

Using BadBoy side by side with SoapUI tests both SIDES Web Services and Web Application at the same time.

### **10.6 *ERWin***

AllFusion® ERwin® Data Modeler (AllFusion ERwin DM) from Computer Associates International, Inc. (CA) is an industry-leading data modeling solution that enables SIDES to create and maintain databases. Data models help visualize data structures so that SIDES can organize, manage and moderate data complexities, database technologies and the deployment environment.