

RIFIDI Engineering

James Percent
james@pramari.com

10/10/2010

Contents

1	Introduction	2
2	Operating System	2
3	Network and Email	3
4	Project Management Software	3
5	Eclipse Setup	6
6	Edge Server	11
6.0.1	Background	11
6.0.2	Edge Server Configuration	11
6.1	Running Unit, System and RegressionTests	13
7	Edge-Server Applications	13
7.1	My Fisher-Price RIFIDI Application	13
7.2	Testing My Fisher-Price RIFIDI Application	13
8	Modifying the Edge Server	13
8.1	Configuration	13
8.2	Deployment	13

List of Figures

1	Pramari Account Settings	4
2	Server Settings	5

3	SMTP Settings	6
4	Eclipse Workspace	7
5	Eclipse Software Installation	8
6	Subclipse	9
7	Additional Eclipse Packages	10
8	Lifecycle of a Bundle	12
9	Switching Perspective	14
10	SVN Exploring Perspective	15
11	Adding a New Repository 1	16
12	Adding a New Repository 2	17
13	Edge Server Source Checkout	18
14	Setting the Target Platform	19
15	Run Configurations	20
16	Default Edge Server Run Configuration	21
17	OSGi Console	22

1 Introduction

This document is the basis for the getting started on the RIFIDI engineering team. It is the single source for all the information you need to get going. It is obviously evolving, so let us know if you find anything missing, outdated, or incorrect.

Rather than litter the entire document with all the links, we have chosen a citation-based reference approach. This keeps the presentation concise and the reference material co-located. Citations should be clickable and any link within a citation should also be directly clickable. This may seem annoying at first, but in the end we think having a concise index to the reference material is highly advantageous and worth a bit of clicking around.

2 Operating System

This section provides information about installing and configuring Ubuntu Linux, which is our supported platform. We should be able to run on any platform that supports Java 6, but such configurations are beyond the scope of this document.

The supported version of Ubuntu is 10.04.1 LTS, Lucid Lynx [1, 2]. Instructions regarding Ubuntu installation and configuration can be found in [3]. If you're not familiar with Ubuntu Linux, please take note of the package management system because it is an important piece of the platform in terms of getting your environment up and running. An old tutorial is here [4], and, as always, Google is your friend.

There are only a few packages required to setup the environment; this command should work:

```
$ sudo apt-get install eclipse emacs23 sun-java6-jdk thunderbird vim.
```

Please note that, the period at the end of the command is a syntactic element of the English language, and it is not part of the command.

3 Network and Email

Our Pramari email accounts are our primary means of communication. All email communication for company related information should happen via Pramari email accounts. Our emails are proprietary and confidential and should be treated with care.

Each user has a soft limit of 100 megabytes of email. We recommend using the Mozilla Thunderbird email client [6]. If you completed Section 2, then you already have Thunderbird installed.

Email can also be accessed through a web client [5, 7]; the server, *harwood.textdrive.com* must be selected from the drop-down list. Your userID is *firstname – pramari*; ask someone for your initial password.

The steps required to configure Thunderbird are straight forward:

- add a new account; and
- use the settings shown in Figures 1- 3, substituting your name and user-id for Prasith Govin and pgovin respectively.

4 Project Management Software

We use Scrum to manage our development projects. There's a plethora of Scrum tutorials, guides and books, but you can probably grok everything you need on-the-job. That said, I recommend [11] as a brief introduction, and the original publication [10] for a more thorough treatment. We have a copy of [10] in the office.

We use basecamp [8] for client-facing project management and documentation, and we use Rally [9] for Sprint planning and tracking (Sprints are the units of work in Scrum). You should have accounts to both of these, so let someone know if you do not.

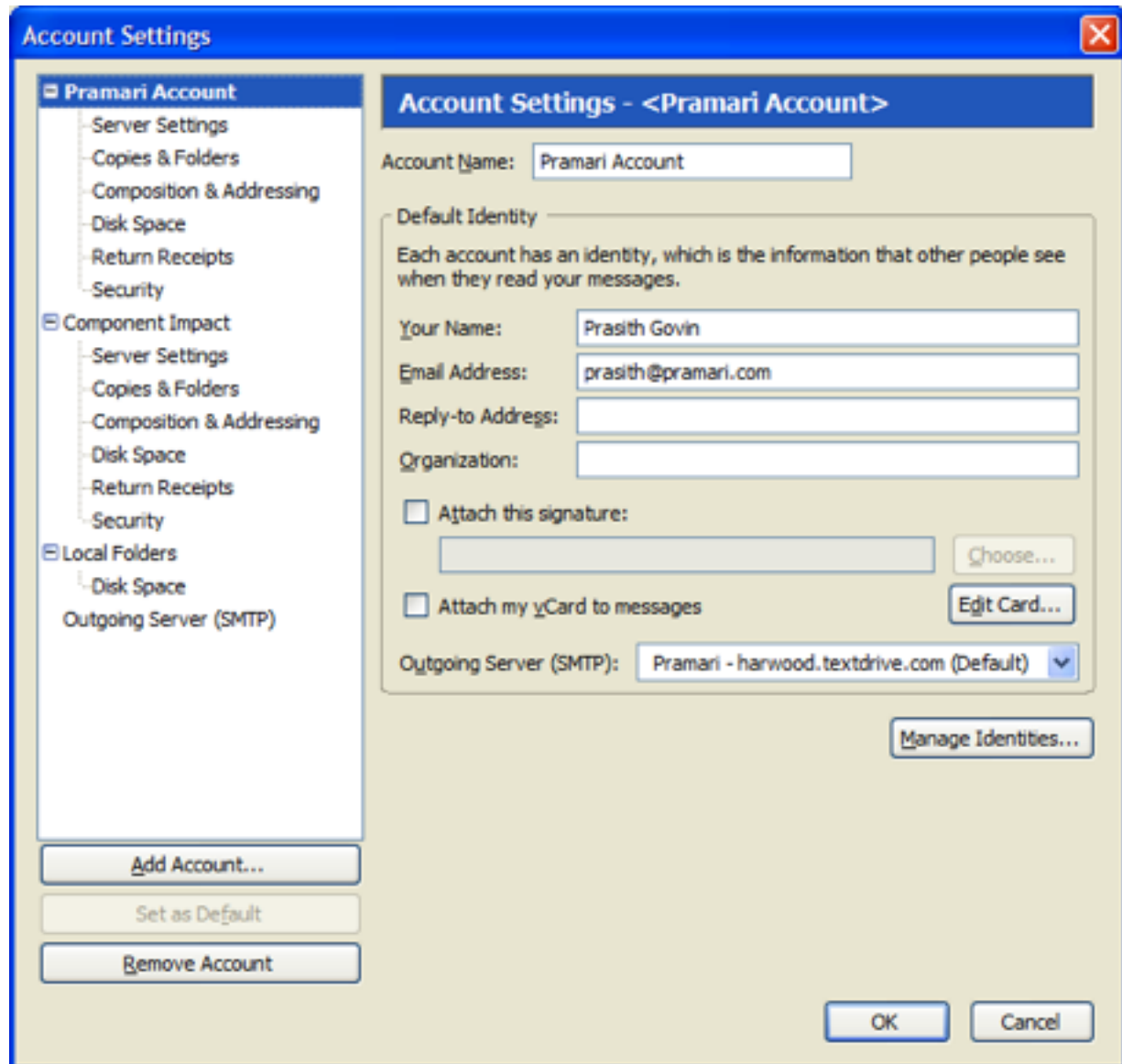


Figure 1: Pramari Account Settings

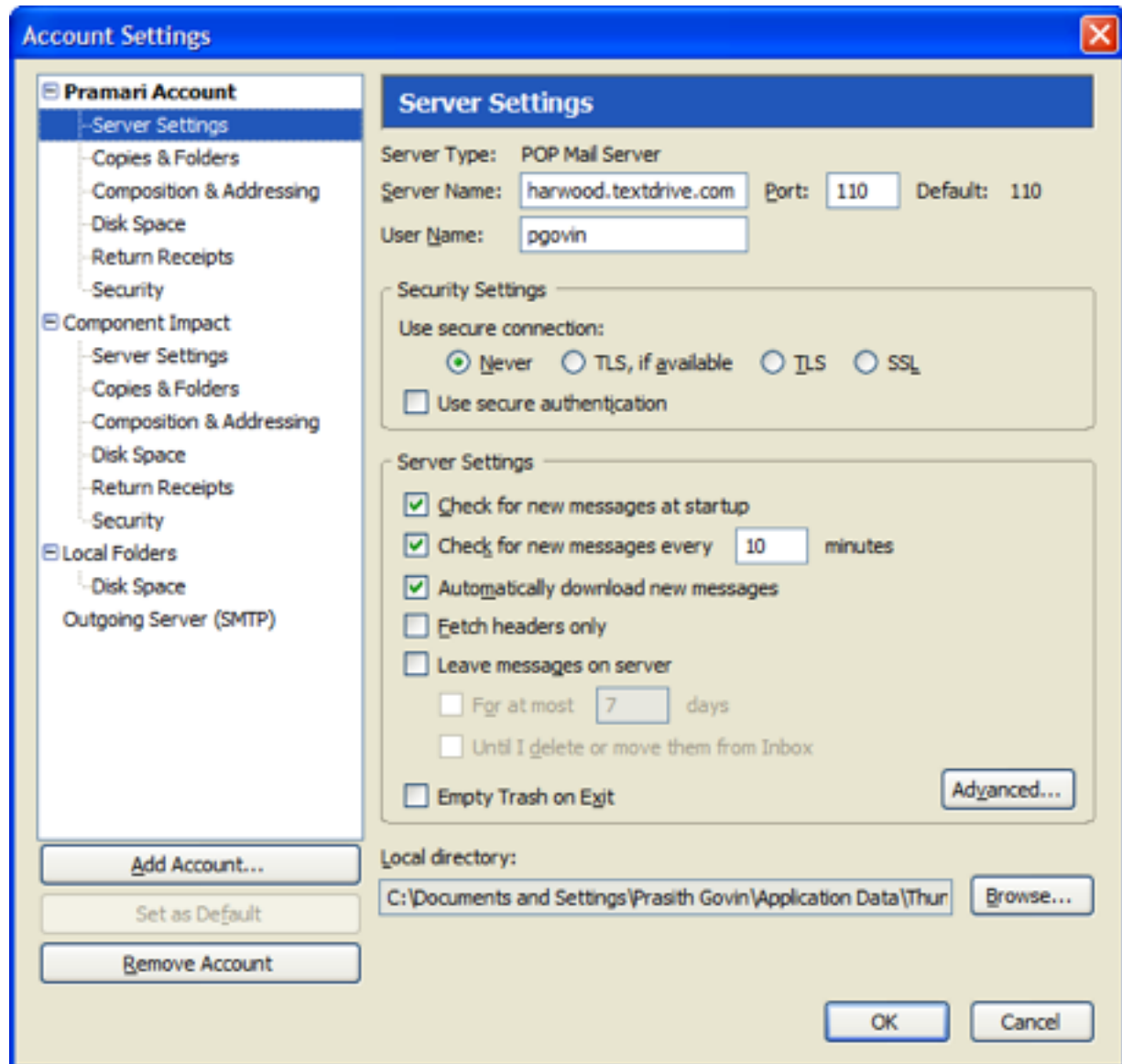


Figure 2: Server Settings



Figure 3: SMTP Settings

5 Eclipse Setup

Our development process, for better or worse, is very tightly coupled to the Eclipse platform [15]. If you completed Section 2, then Eclipse should be installed on your system. From the Ubuntu desktop, click: *Applications*→*Development*→*Eclipse*. We recommend creating a workspace for each project, as depicted in Figure 4.

All RIFIDI-based software can be downloaded from our online repositories [12, 13, 14]. We use Subclipse [22] to interface with our SVN server. To install Subclipse, from within Eclipse, click: *Help*→*Install new software* as shown in Figure 5. A dialog box will appear; click on add and paste in the Subclipse update site URL [25]; check all required boxes and click finish as shown in Figure 6. Subclipse’s documentation is installed with the plugin and can be accessed from Eclipse’s help menu. An online version can be found at [24].

We depend on several other packages: *Eclipse Plug-in Development Environment*, *PDE/API Tools Environment*, *Java Development Tools*, *Eclipse XML Editors and Tools*, *JavaScript Developer Tools*, *Web, XML and Java EE Development*. To install these packages, click: *Help*→*Install new software*. From the drop-down box select the Galileo update site [21] and select the packages as shown in Figure 7.

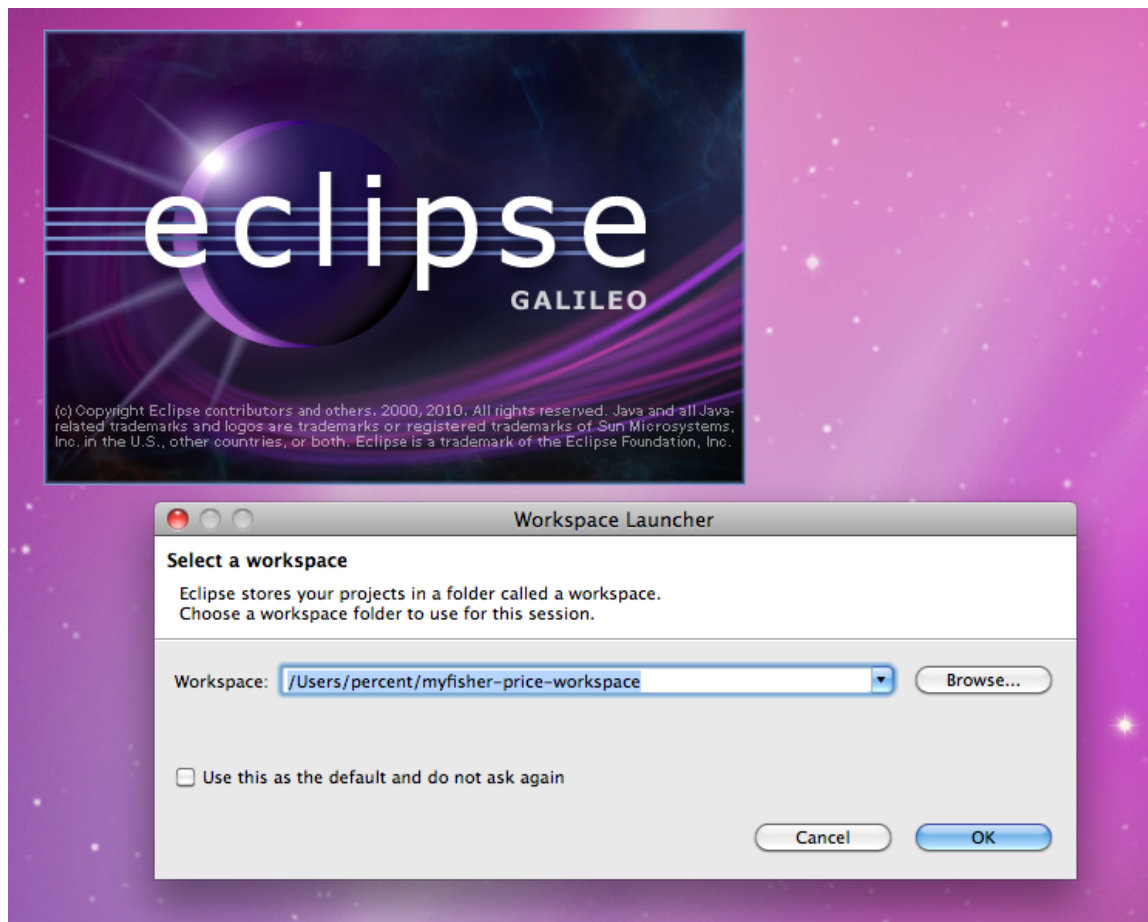


Figure 4: Eclipse Workspace

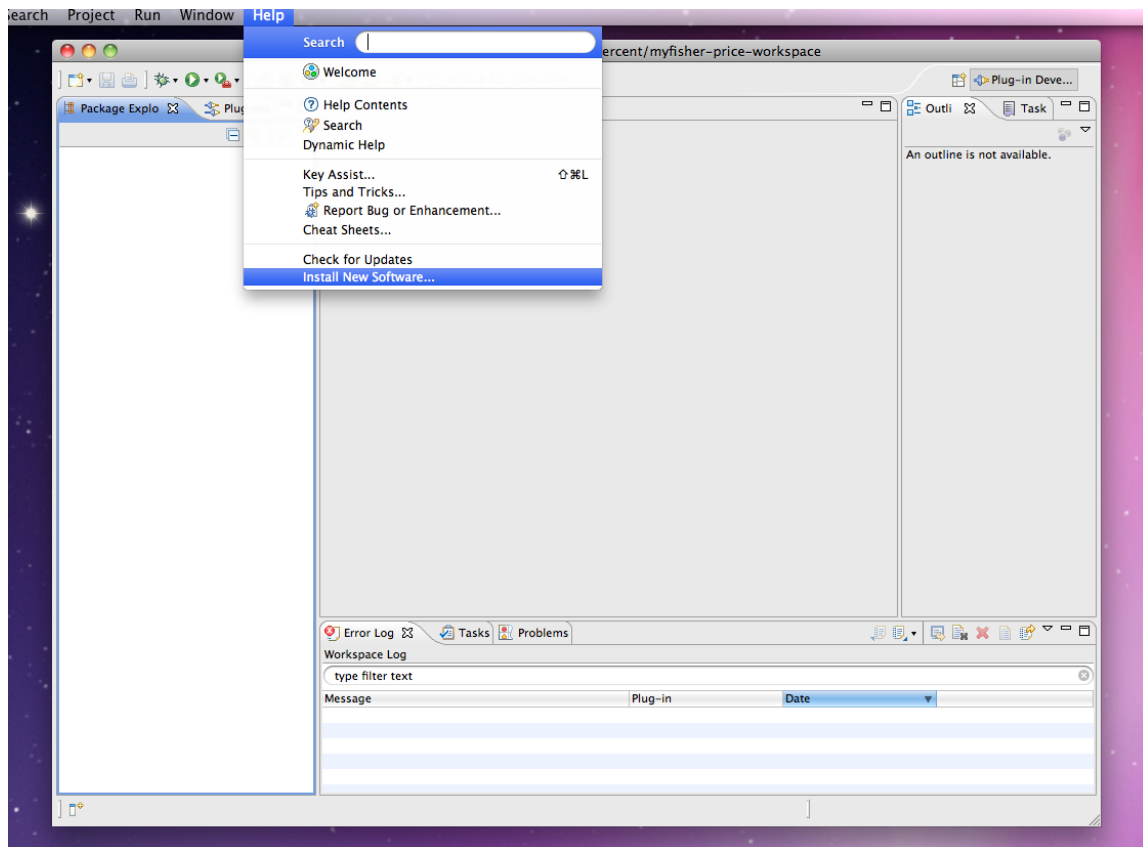


Figure 5: Eclipse Software Installation

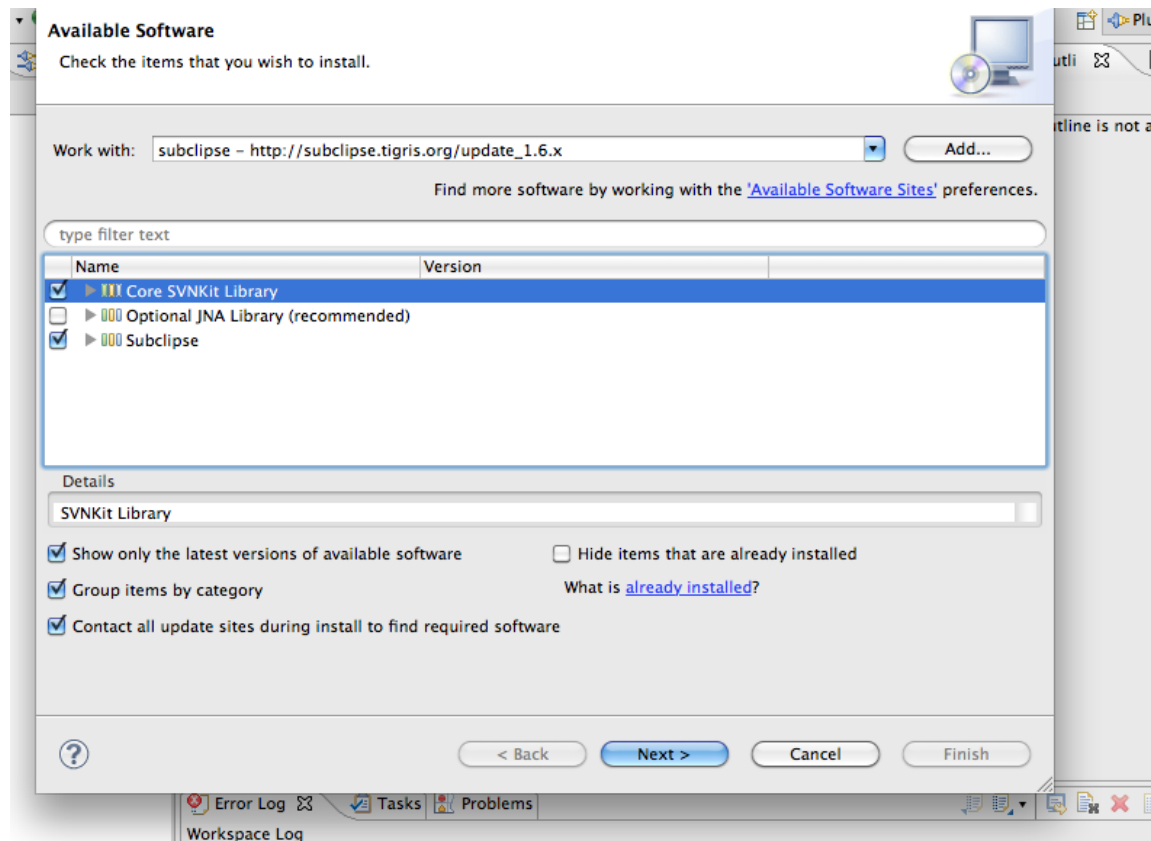


Figure 6: Subclipse

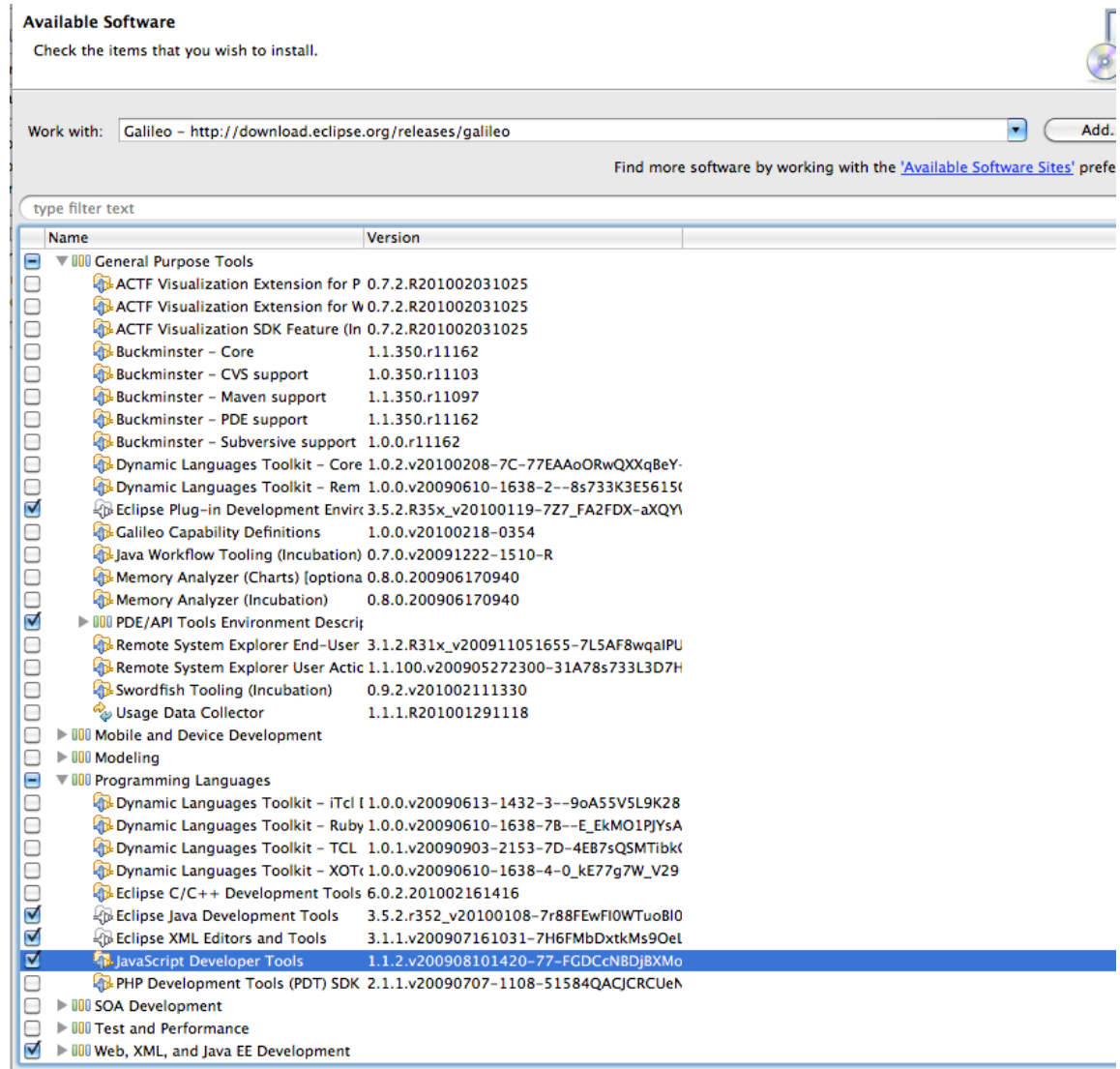


Figure 7: Additional Eclipse Packages

6 Edge Server

6.0.1 Background

The Edge Server is an OSGi-based service platform. Our development is done within the Eclipse Plug-in Development Environment (PDE) [16]. PDE provides OSGi tooling support and is based on Equinox [18], which is the reference implementation of the OSGi Alliance [19].

OSGi is essentially a dynamic module framework for Java. The fundamental unit of decomposition in OSGi-based systems is the bundle; everything is encapsulated by a bundle.

A bundle is basically a collection of packages and compile and run time dependancies. Each bundle has a manifest file which is where these dependancies are specified. The bundle's manifest file contains its unique identification, which includes its version, and specifies the packages it imports and exports.

The bundle lifecycle state-machine is depicted in Figure 8. When the OSGi run-time is started every available bundle is installed. If all the dependencies of a given bundle are met, then the bundle is resolved and it can be started. A bundle must be stopped to be uninstalled or restarted. Bundles are automatically stopped by the OSGi runtime when it is shutdown.

An essential part of developing PDE services is the target platform. The target platform defines all the bundles that a PDE-based service can use. Therefore, it contains all the OSGi bundles as well as any other third-party bundles that are used. PDE provides a basic target platform, but the Edge Server extends it with many third-party bundles. The target platform is well documented in the help chapter: *Plug-in Development Environment Guide*.

Dependency injection [23] is a design paradigm whereby the logic of the system is separated from acquiring and configuring dependancies. The Edge Server relies on Spring Dynamic Modules [20] to register services in the OSGi service registry and to handle dependency injection. Each bundle has a Spring directory that defines its services and dependency injection rules.

6.0.2 Edge Server Configuration

Now we are ready to checkout the Edge Server sources. To accomplish this we need to switch to the SVN Repository Exploring perspective, as shown in Figures 9 and 10.

Next, we need to add the Edge Server repository [12]. From within the SVN Repositories tab, right-click and select: *New→Repository Location* as shown in Figure 11. A dialog-box

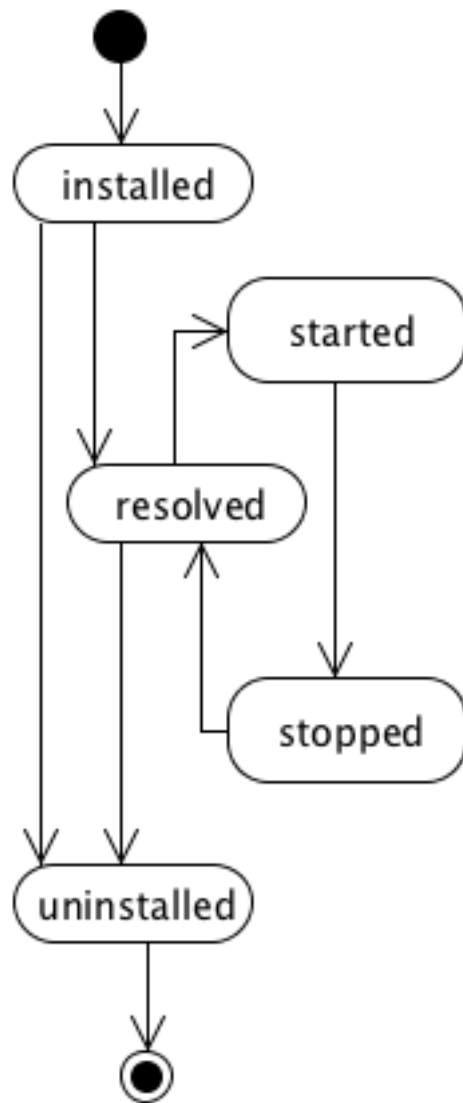


Figure 8: Lifecycle of a Bundle

will appear and the URL from [12] needs to be pasted into it, as shown in Figure 12.

Next, we need to checkout the Edge Server sources. From the Exploring perspective click: *rifidi*→*trunk*. Select the following folders: *org.rifidi.app.example*, *org.rifidi.app.template*, *org.rifidi.edge.api*, *org.rifidi.edge.app.diag*, *org.rifidi.edge.app.tracking*, *org.rifidi.edge.console*, *org.rifidi.edge.core*, *org.rifidi.edge.init* and *Rifidi-SDK*. Subsequently right-click and select *Checkout*. Figure 13 shows you what this looks like. This takes about 10 - 30 minutes to complete depending on connection speeds. Now would be a good time to grab some coffee.

After the download completes, return to the Java perspective, so that we can set the target platform. To set the target platform, from within the package explorer panel, open *Rifidi-SDK*→*RifidiHome*→*org.rifidi.edge.target.target*. This should come up in a special Eclipse target-platform-editor. Click *Set as Target Platform* as shown in Figure 14.

Finally we need to setup the run configuration. We have included a *EdgeServer.launch* file in the *Rifidi-SDK*, so an Eclipse run-configuration already exists. Open the run configuration as shown in Figure 15, and find the *Edge Server* run configuration under *OSGI Framework* as show in Figure 16, and click run. To confirm the system is up and running, from the Eclipse console, type apps as shown in Figure 17.

6.1 Running Unit, System and RegressionTests

7 Edge-Server Applications

7.1 My Fisher-Price RIFIDI Application

7.2 Testing My Fisher-Price RIFIDI Application

8 Modifying the Edge Server

8.1 Configuration

8.2 Deployment

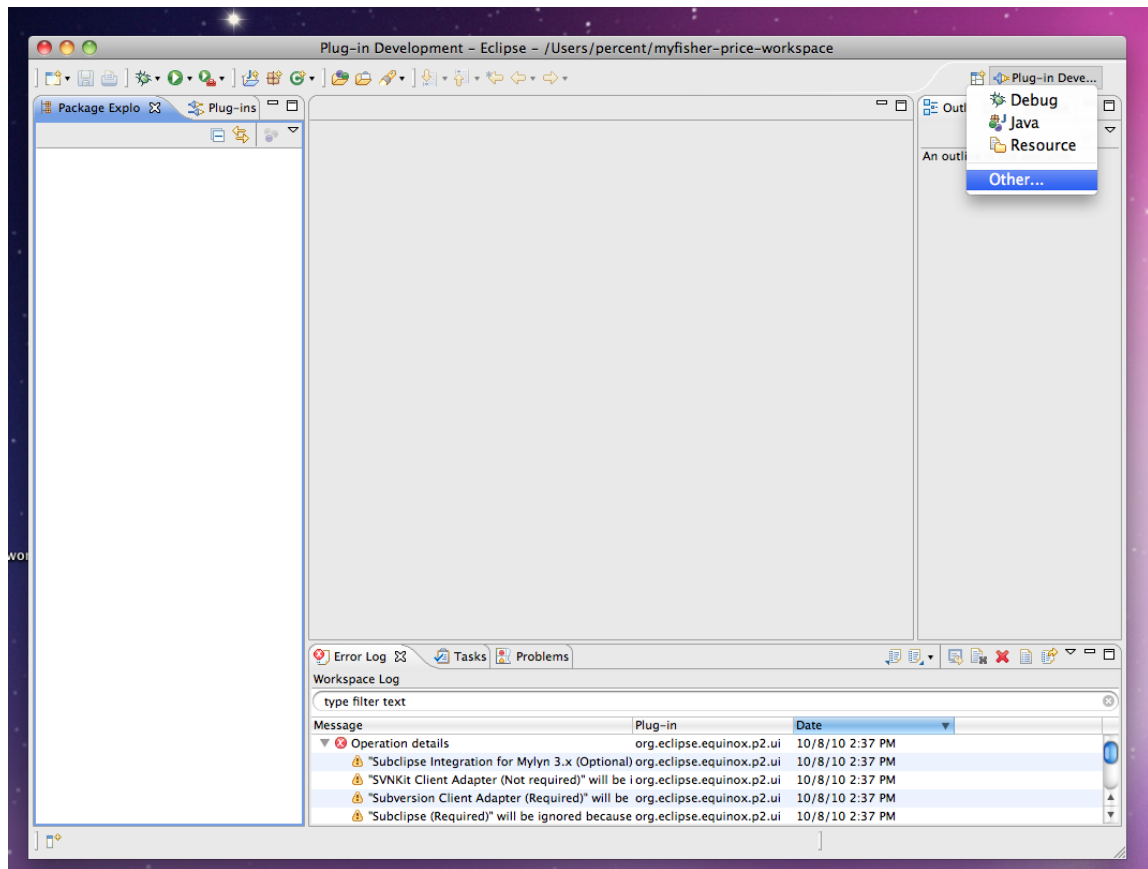


Figure 9: Switching Perspective

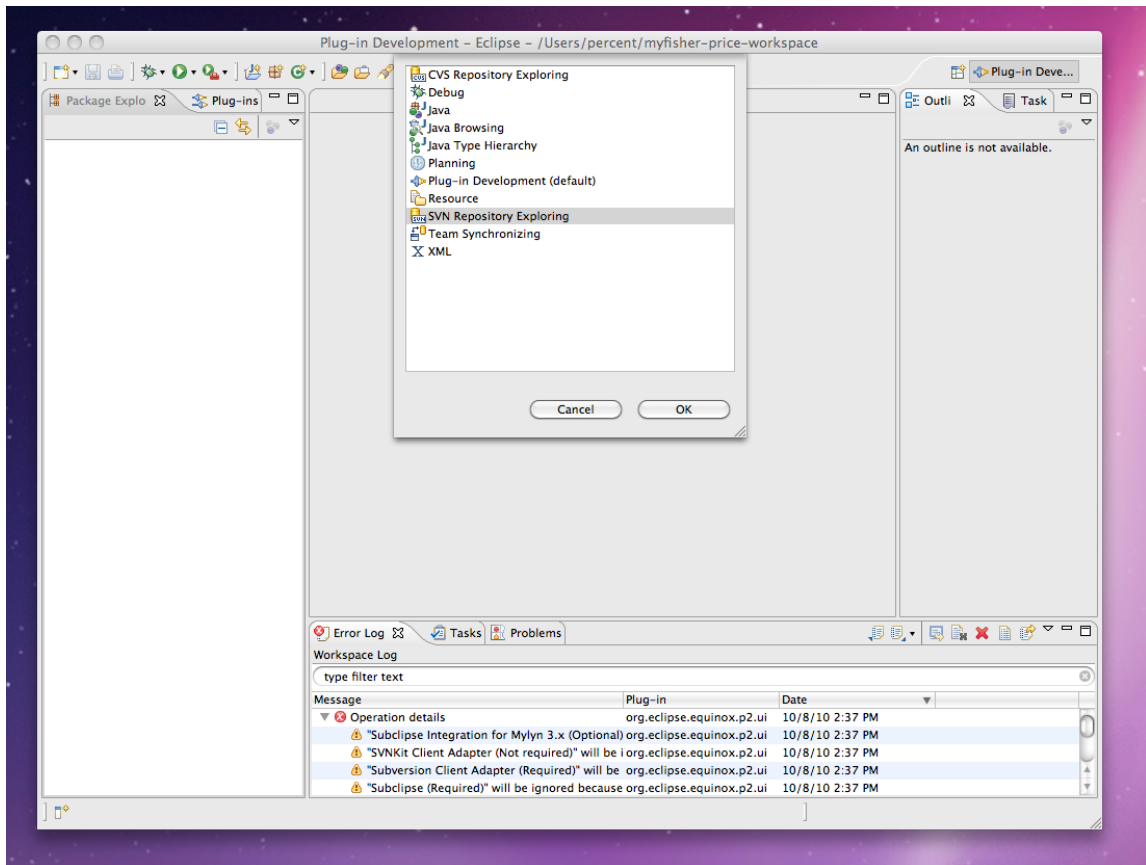


Figure 10: SVN Exploring Perspective

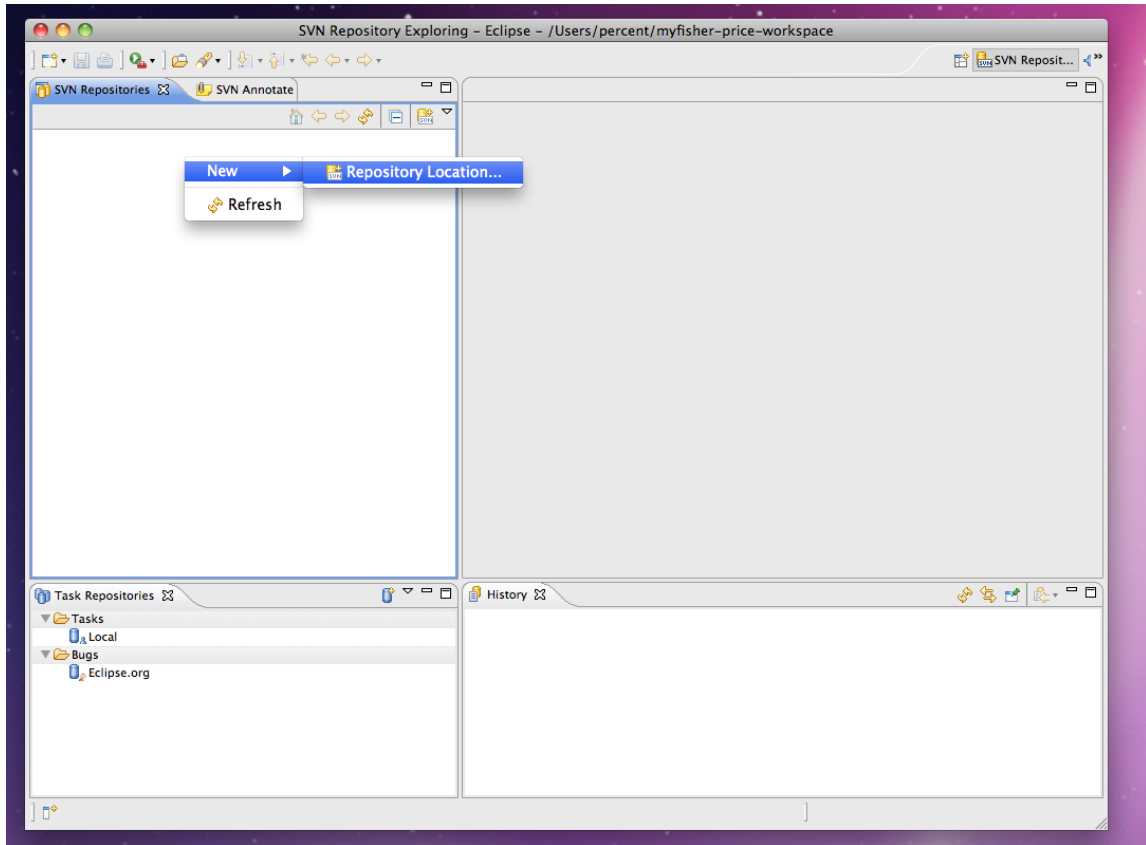


Figure 11: Adding a New Repository 1

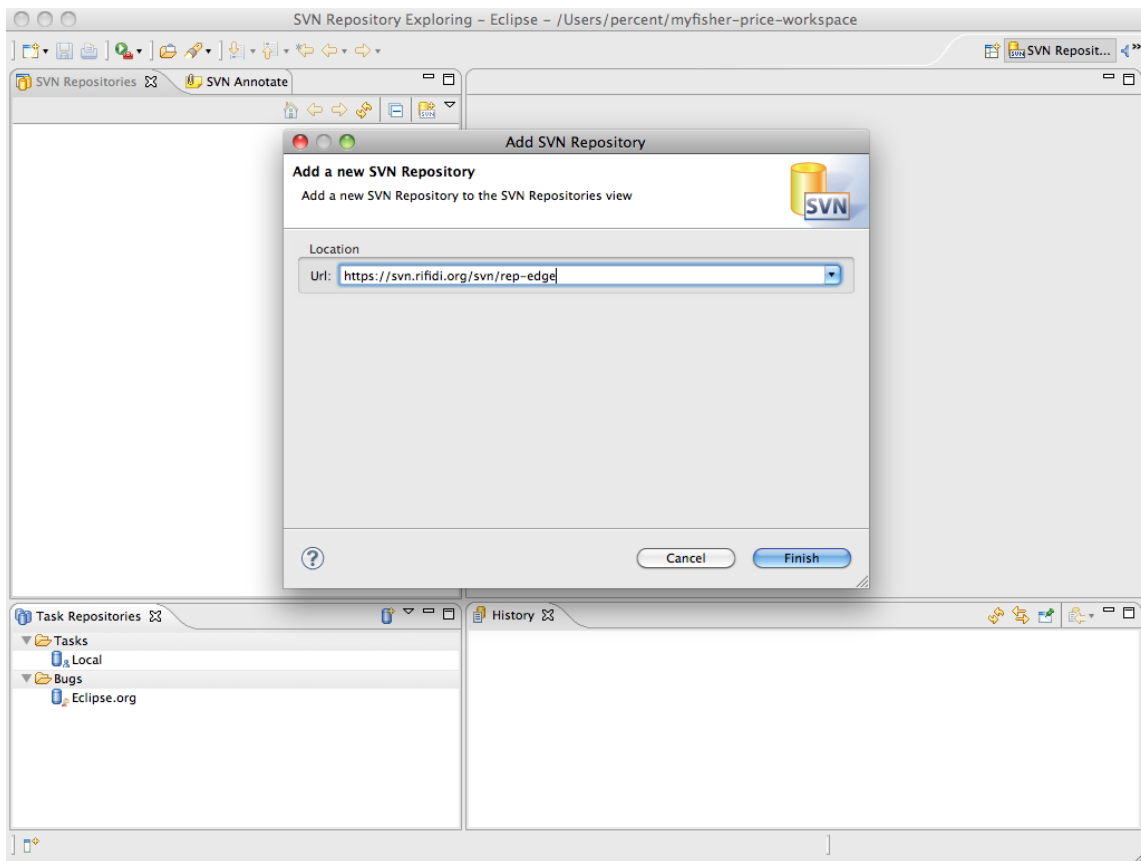
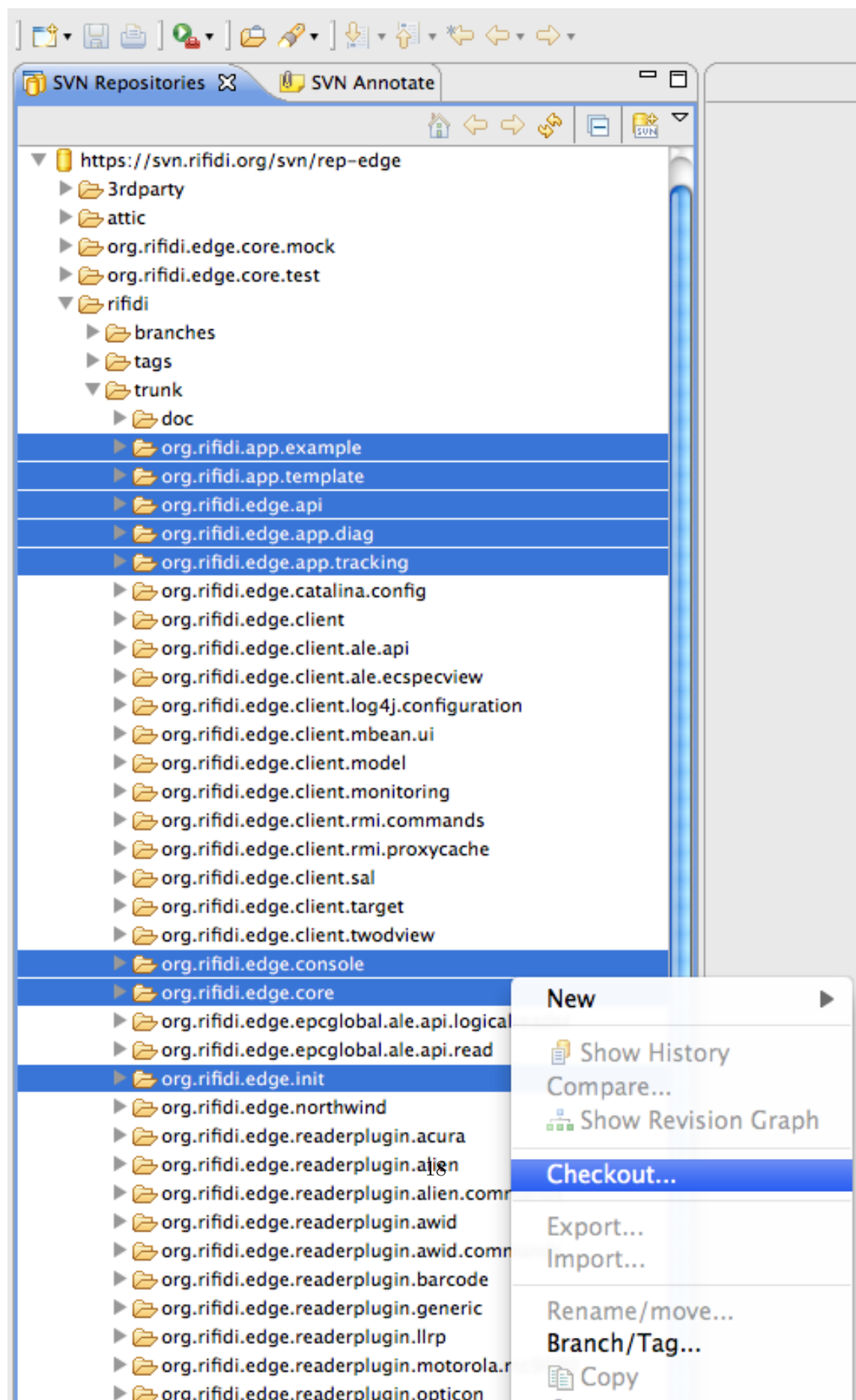


Figure 12: Adding a New Repository 2



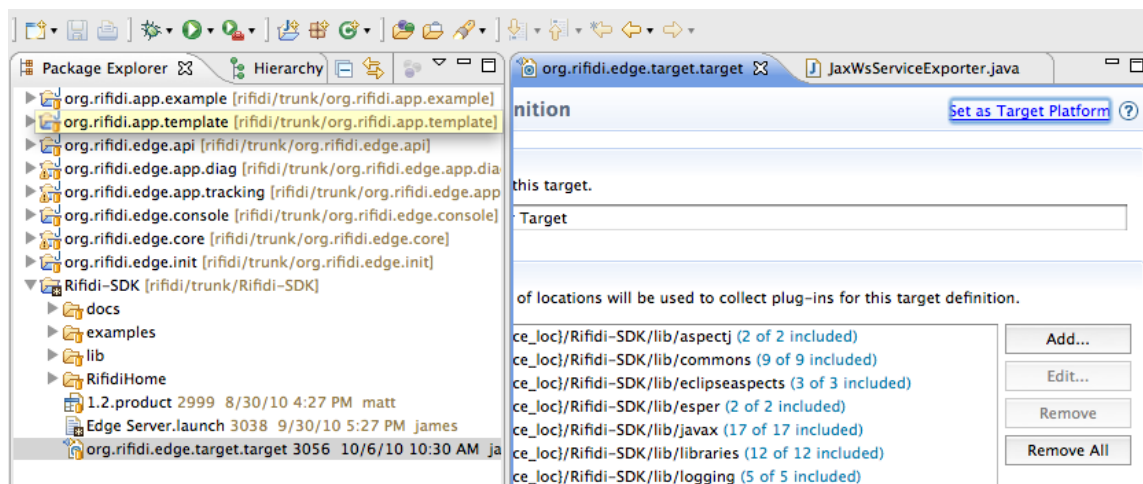


Figure 14: Setting the Target Platform

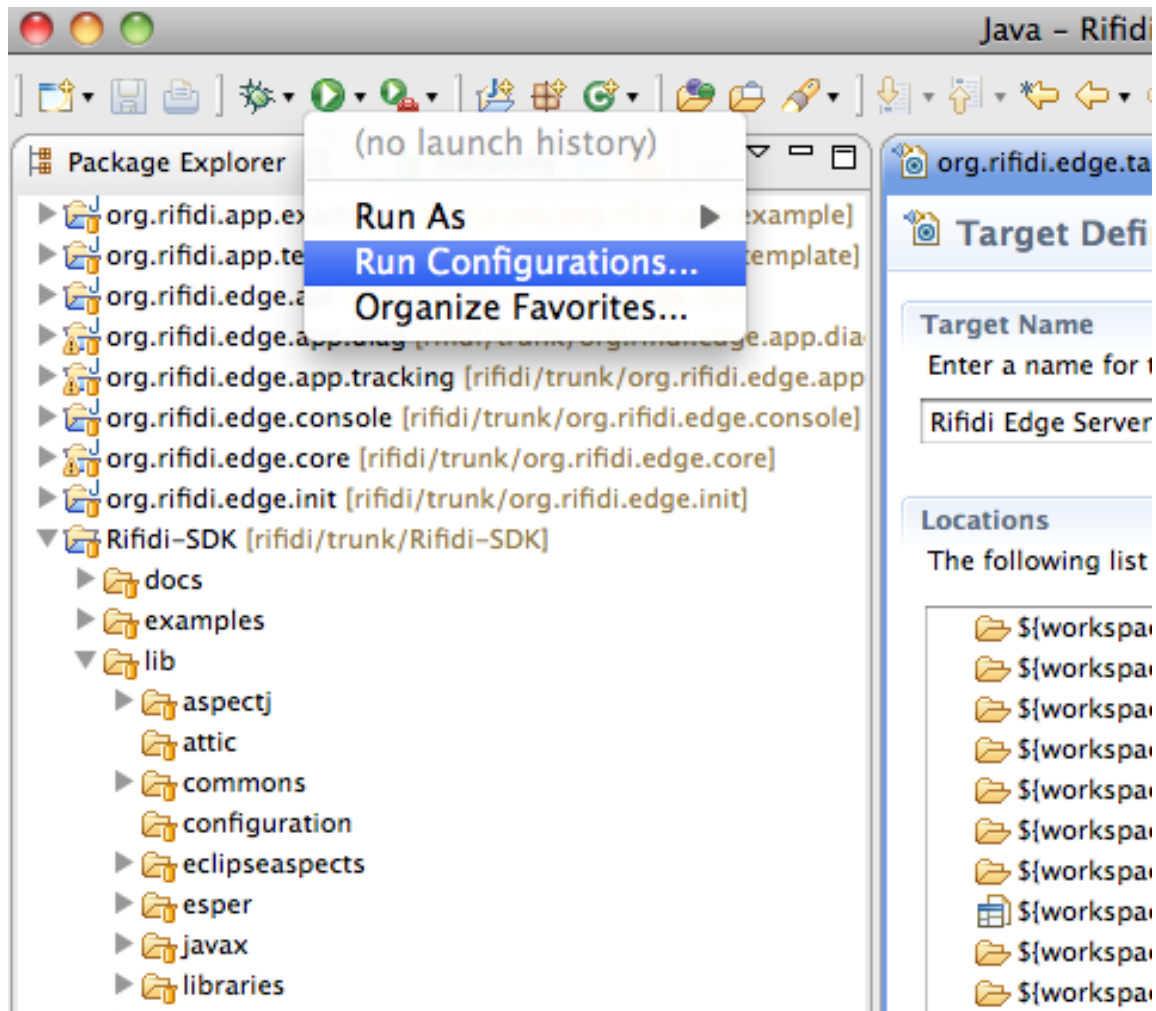


Figure 15: Run Configurations

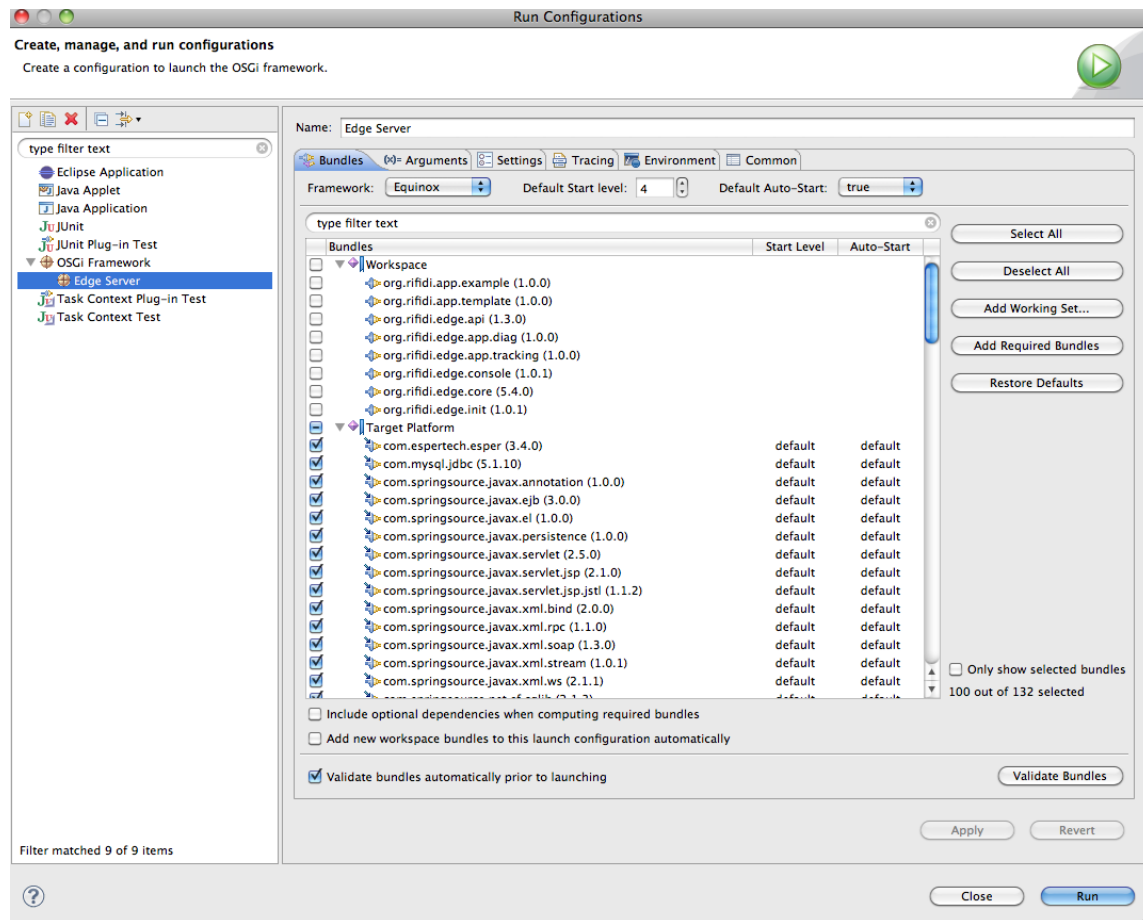
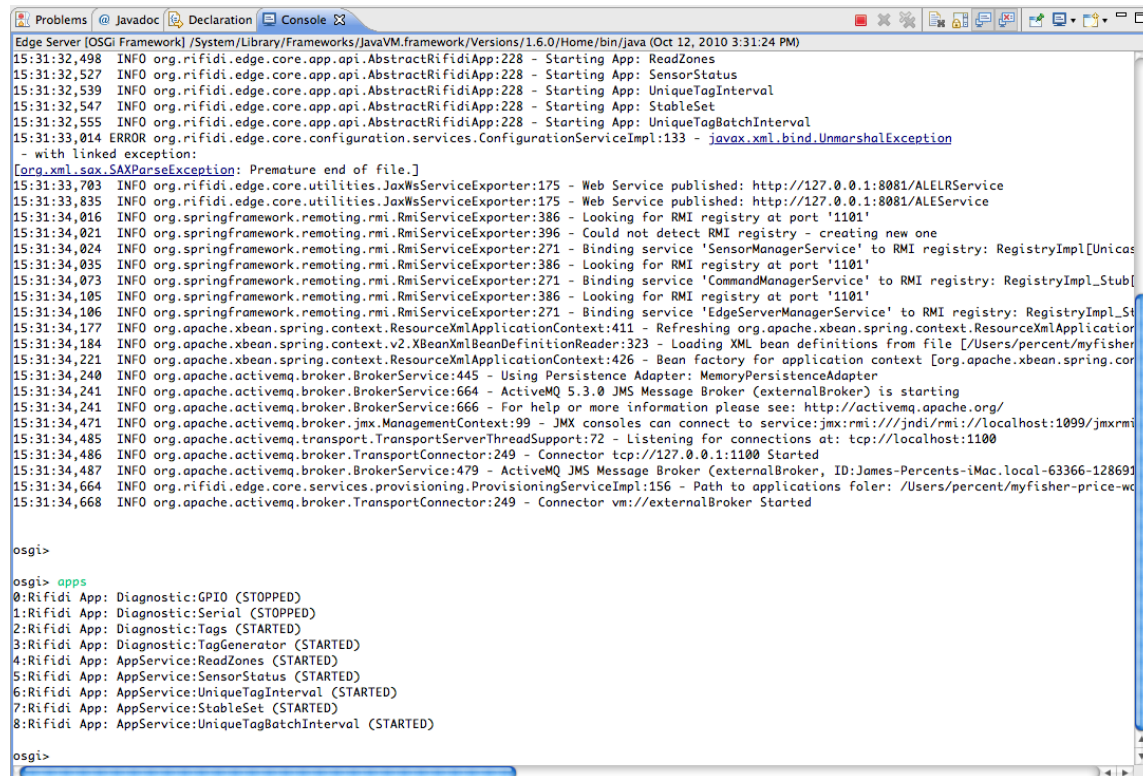


Figure 16: Default Edge Server Run Configuration



```
Edge Server [OSGi Framework] /System/Library/Frameworks/JavaVM.framework/Versions/1.6.0/Home/bin/java (Oct 12, 2010 3:31:24 PM)
15:31:32,498 INFO org.rifidi.edge.core.app.api.AbstractRifidiApp:228 - Starting App: ReadZones
15:31:32,527 INFO org.rifidi.edge.core.app.api.AbstractRifidiApp:228 - Starting App: SensorStatus
15:31:32,539 INFO org.rifidi.edge.core.app.api.AbstractRifidiApp:228 - Starting App: UniqueTagInterval
15:31:32,547 INFO org.rifidi.edge.core.app.api.AbstractRifidiApp:228 - Starting App: StableSet
15:31:32,555 INFO org.rifidi.edge.core.app.api.AbstractRifidiApp:228 - Starting App: UniqueTagBatchInterval
15:31:33,014 ERROR org.rifidi.edge.core.configuration.services.ConfigurationServiceImpl:133 - javax.xml.bind.UnmarshalException
- with linked exception:
[org.xml.sax.SAXParseException: Premature end of file.]
15:31:33,703 INFO org.rifidi.edge.core.utilities.JaxWsServiceExporter:175 - Web Service published: http://127.0.0.1:8081/ALELRService
15:31:33,835 INFO org.rifidi.edge.core.utilities.JaxWsServiceExporter:175 - Web Service published: http://127.0.0.1:8081/ALEService
15:31:34,016 INFO org.springframework.remoting.rmi.RmiServiceExporter:386 - Looking for RMI registry at port '1101'
15:31:34,021 INFO org.springframework.remoting.rmi.RmiServiceExporter:396 - Could not detect RMI registry - creating new one
15:31:34,024 INFO org.springframework.remoting.rmi.RmiServiceExporter:271 - Binding service 'SensorManagerService' to RMI registry: RegistryImpl[Unicas
15:31:34,035 INFO org.springframework.remoting.rmi.RmiServiceExporter:386 - Looking for RMI registry at port '1101'
15:31:34,073 INFO org.springframework.remoting.rmi.RmiServiceExporter:271 - Binding service 'CommandManagerService' to RMI registry: RegistryImpl_Stub[
15:31:34,105 INFO org.springframework.remoting.rmi.RmiServiceExporter:386 - Looking for RMI registry at port '1101'
15:31:34,106 INFO org.springframework.remoting.rmi.RmiServiceExporter:271 - Binding service 'EdgeServerManagerService' to RMI registry: RegistryImpl_St
15:31:34,177 INFO org.apache.xbean.spring.context.ResourceXmlApplicationContext:411 - Refreshing org.apache.xbean.spring.context.ResourceXmlApplication
15:31:34,184 INFO org.apache.xbean.spring.context.v2.XBeanXmlBeanDefinitionReader:323 - Loading XML bean definitions from file [/Users/percent/myfisher
15:31:34,221 INFO org.apache.xbean.spring.context.ResourceXmlApplicationContext:426 - Bean factory for application context [org.apache.xbean.spring.cor
15:31:34,240 INFO org.apache.activemq.broker.BrokerService:445 - Using Persistence Adapter: MemoryPersistenceAdapter
15:31:34,241 INFO org.apache.activemq.broker.BrokerService:664 - ActiveMQ 5.3.0 JMS Message Broker (externalBroker) is starting
15:31:34,241 INFO org.apache.activemq.broker.BrokerService:666 - For help or more information please see: http://activemq.apache.org/
15:31:34,471 INFO org.apache.activemq.broker.jmx.ManagementContext:99 - JMX consoles can connect to service:jmx:rmi:///jndi/rmi://localhost:1099/jmxrmi
15:31:34,485 INFO org.apache.activemq.transport.TransportServerThreadSupport:72 - Listening for connections at: tcp://localhost:1100
15:31:34,486 INFO org.apache.activemq.broker.TransportConnector:249 - Connector tcp://127.0.0.1:1100 Started
15:31:34,487 INFO org.apache.activemq.broker.BrokerService:479 - ActiveMQ JMS Message Broker (externalBroker, ID:James-Percents-iMac.local-63366-128691
15:31:34,664 INFO org.rifidi.edge.core.services.provisioning.ProvisioningServiceImpl:156 - Path to applications folder: /Users/percent/myfisher-price-wa
15:31:34,668 INFO org.apache.activemq.broker.TransportConnector:249 - Connector vm://externalBroker Started

osgi>

osgi> apps
0:Rifidi App: Diagnostic:GPIO (STOPPED)
1:Rifidi App: Diagnostic:Serial (STOPPED)
2:Rifidi App: Diagnostic:Tags (STARTED)
3:Rifidi App: Diagnostic:TagGenerator (STARTED)
4:Rifidi App: AppService:ReadZones (STARTED)
5:Rifidi App: AppService:SensorStatus (STARTED)
6:Rifidi App: AppService:UniqueTagInterval (STARTED)
7:Rifidi App: AppService:StableSet (STARTED)
8:Rifidi App: AppService:UniqueTagBatchInterval (STARTED)

osgi>
```

Figure 17: OSGi Console

References

- [1] Ubuntu Releases. <http://releases.ubuntu.com>.
- [2] Ubuntu 10.04.1. <http://releases.ubuntu.com/lucid>.
- [3] Ubuntu 10.04.1 Documentation. <https://help.ubuntu.com/10.04/index.html>.
- [4] Apt. <http://www.debian.org/doc/manuals/apt-howto/ch-apt-get.en.html>.
- [5] Webmail. <http://webmail.textdrivehosting.com/index.php>.
- [6] Thunderbird. <http://www.mozilla.org/projects/thunderbird>.
- [7] Mail server `harwood.textdrive.com`.
- [8] Pramari Basecamp. <https://pramari.basecamphq.com>.
- [9] Rally. <https://community.rallydev.com>.
- [10] Schwaber, Ken. Agile Project Management with Scrum.
- [11] Scrum Guide. <http://www.scrum.org/scrumguideenglish>.
- [12] RIFIDI Edge Server. <https://svn.rifidi.org/svn/rep-edge>.
- [13] RIFIDI Toolkit. <https://svn.rifidi.org/svn/rep-external>.
- [14] Ambient Project. <https://svn.rifidi.org/svn/rep-ambient>.
- [15] Eclipse. <http://www.eclipse.org..>
- [16] Plug-in Development Environment. <http://www.eclipse.org/pde/>.
- [17] OSGi and Equinox: Creating Highly Modular Java Systems. <http://equinoxosgi.org>.
- [18] Equinox. <http://www.eclipse.org/equinox>.
- [19] OSGi Alliance. <http://www.osgi.org/Main/HomePage>.
- [20] Spring Dynamic Modules for OSGi Service Platforms. <http://www.springsource.org/osgi>.
- [21] Eclipse Galileo. <http://download.eclipse.org/releases/galileo>
- [22] Subclipse. <http://subclipse.tigris.org/>.
- [23] Fowler, M. Dependency Injection. <http://martinfowler.com/articles/injection.html>.

- [24] Subclipse online documentation. <http://svn.collab.net/subclipse/help/index.jsp>.
- [25] Subclipse. http://subclipse.tigris.org/update_1.6.x.