

Petals Service Platform

Administration plug-in

Draft

Maxime CARPENTIER

08 / 16 / 2005

Summary

1	PETALS – ADMINISTRATION	3
	WHAT IS THIS PLUG-IN FOR ?	3
	INSTALLATION	4
	ESB PERSPECTIVE	4
	KNOWN BUGS	7
2	PETALS – PLUG-IN CODE	8
	TECHNICAL CHOICES	8
	GENERAL OVERVIEW	8

Illustrations

<i>Figure 1 : Life cycle of a component</i>	3
<i>Figure 2 : Plug-ins list in Eclipse</i>	4
<i>Figure 3 : Main view</i>	5
<i>Figure 4 : Outline view</i>	6
<i>Figure 5 : Zoom view</i>	6
<i>Figure 6 : Property view</i>	7
<i>Figure 7 : right-click on a component</i>	7
<i>Figure 8 : Models</i>	8

1 Petals – Administration

What is this plug-in for?

General purpose

This plug-in allows you to connect, administrate and consult the properties of your Petals Service Platform (also called ‘ESB’ in this document). In a few mouse clicks you can easily control (start, stop, shutdown...) components of a JBI container like the binding and engine components, and the system services which compose the containers.

You can view the whole network of connected Petals containers (the ESB).

More in details...

Your Enterprise Service Bus is made of containers hosting different types of components. For now there are three kinds of components: Binding components, Engine Components and System services; all those components respects the **Java Business Integration** specification.

This application provides three kind of service for administrate the Petals Service Platform:

-Admin service

You can start, stop and shutdown components.

Here is a figure showing a component life-cycle:

A running component is fully functional to provide its services.

A stopped component can’t provide any service, but can be restarted.

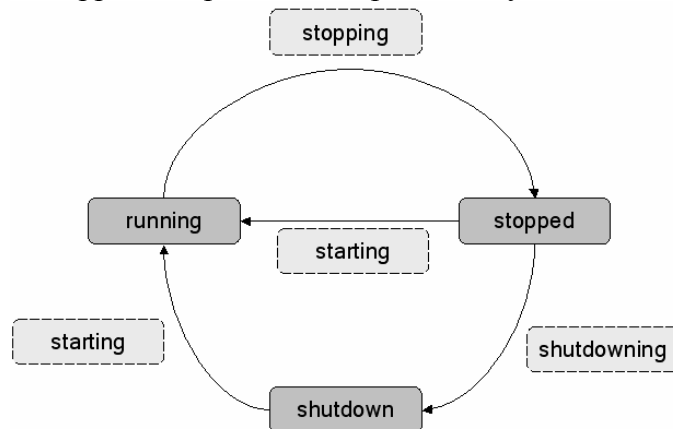


Figure 1 : Life cycle of a component

-Installation service (not yet implemented)

In a few clicks you can remotely install a component in a container.

-Deployment service (not yet implemented)

As for the installation service, you can easily deploy artefacts for a given JBI component.

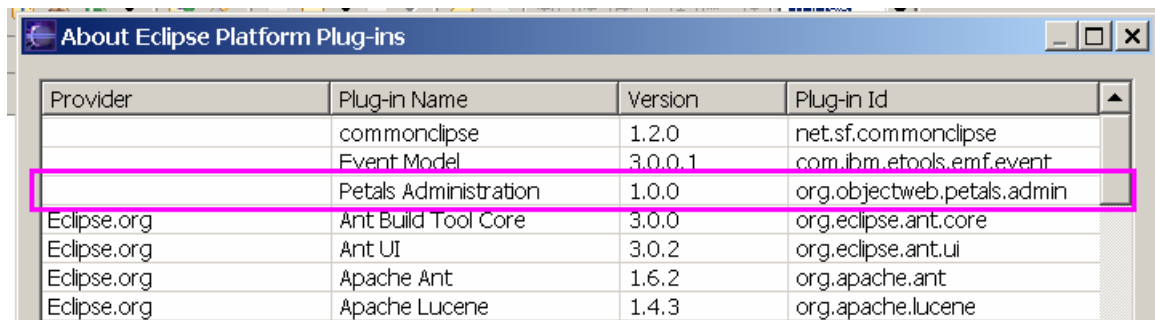
Installation

The application is distributed as an Eclipse plug-in: you can find last version of eclipse at www.eclipse.org

You also need GEF and draw2D plug-ins: www.eclipse.org/gef/

To install the plug-in, just unzip the downloaded file to your Eclipse plug-in directory, then start Eclipse.

You should see a new installed plug-in in the plug-in details (go to Help -> About Eclipse -> Plug-ins details)



Provider	Plug-in Name	Version	Plug-in Id
	commonclipse	1.2.0	net.sf.commonclipse
	Event Model	3.0.0.1	com.ibm.etools.emf.event
	Petals Administration	1.0.0	org.objectweb.petals.admin
Eclipse.org	Ant Build Tool Core	3.0.0	org.eclipse.ant.core
Eclipse.org	Ant UI	3.0.2	org.eclipse.ant.ui
Eclipse.org	Apache Ant	1.6.2	org.apache.ant
Eclipse.org	Apache Lucene	1.4.3	org.apache.lucene

Figure 2 : Plug-ins list in Eclipse

Now, create a new file in your project with the “esb” extension; for example “test.esb”.

Edit the file.

A dialog box pops up and explains how to connect to the Petals Service Platform: you have to define the host and the port of the global JNDI server defined in the whole Petals Service Platform configuration (see *Petals-Use* document for further information about configuration).

ESB perspective

Main view

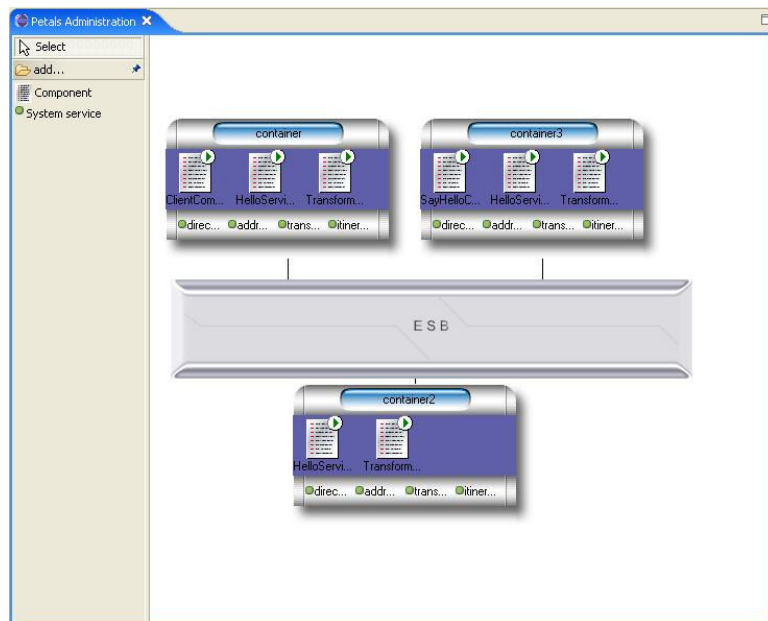


Figure 3 : Main view

The main view allows you to visualize the structure of your ESB. An ESB is composed of containers, binding and engine components, system services and the bus.

Outline view

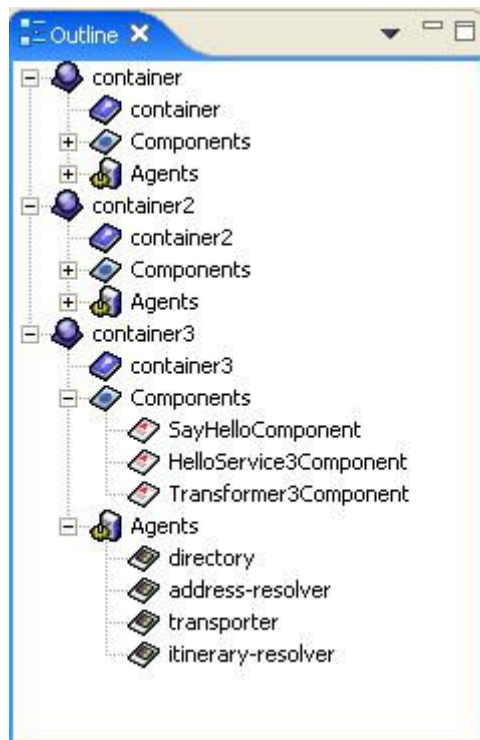


Figure 4 : Outline view

The Outline view is a tree view of the ESB structure.

Zoom view

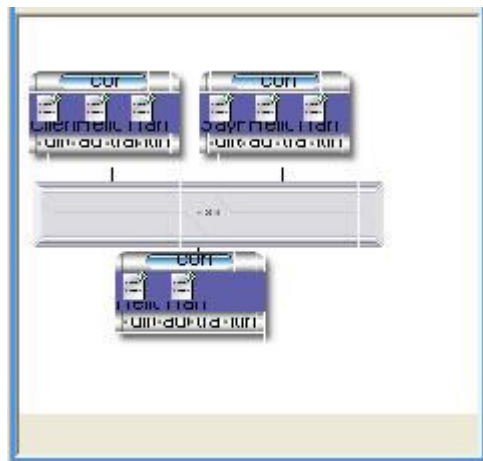
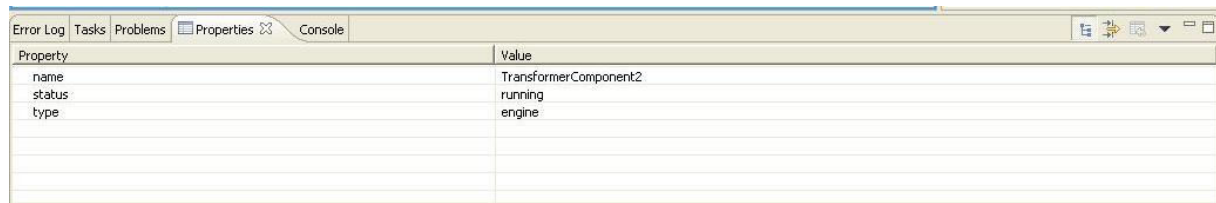


Figure 5 : Zoom view

The Zoom view allows you to see a thumb view of the ESB.

Property view



Property	Value
name	TransformerComponent2
status	running
type	engine

Figure 6 : Property view

The property view is the view by which you can edit each components property: at first launch, you have to set the server address which is a property of the bus component

Interacting

You can easily interact with the view: just right-click on a component to administrate it.



Figure 7 : right-click on a component

You can also set up the graphical disposition of various components: for containers, hold left mouse button to on their shadows to move the component.

Note: The bus can not be moved.

Known Bugs

- The address of the connection to your ESB is automatically saved on change, there's no undo system implemented yet
- There's a graphical bug when selecting a component inside a container and scrolling up or down the component contents
- There's a memory overflow sometimes which makes Eclipse crash.

2 Petals – Plug-in code

Technical choices

For this plug-in i used GEF and draw2D which can be found at www.eclipse.org/gef/
Remote administration has been set using JMX : java.sun.com/products/JavaManagement/

General overview

This plug-in has been made using the GEF MVC architecture: see GEF documentation if you want further information about how it works.

The application is divided into several packages:

org.objectweb.petals.admin

the root package : contains main plug-in files like the Editor EsbEditor.java

org.objectweb.petals.admin.models

Each graphical component has a model which contains data about its references for JMX administration and about its grapical representation (size,location, etc...)

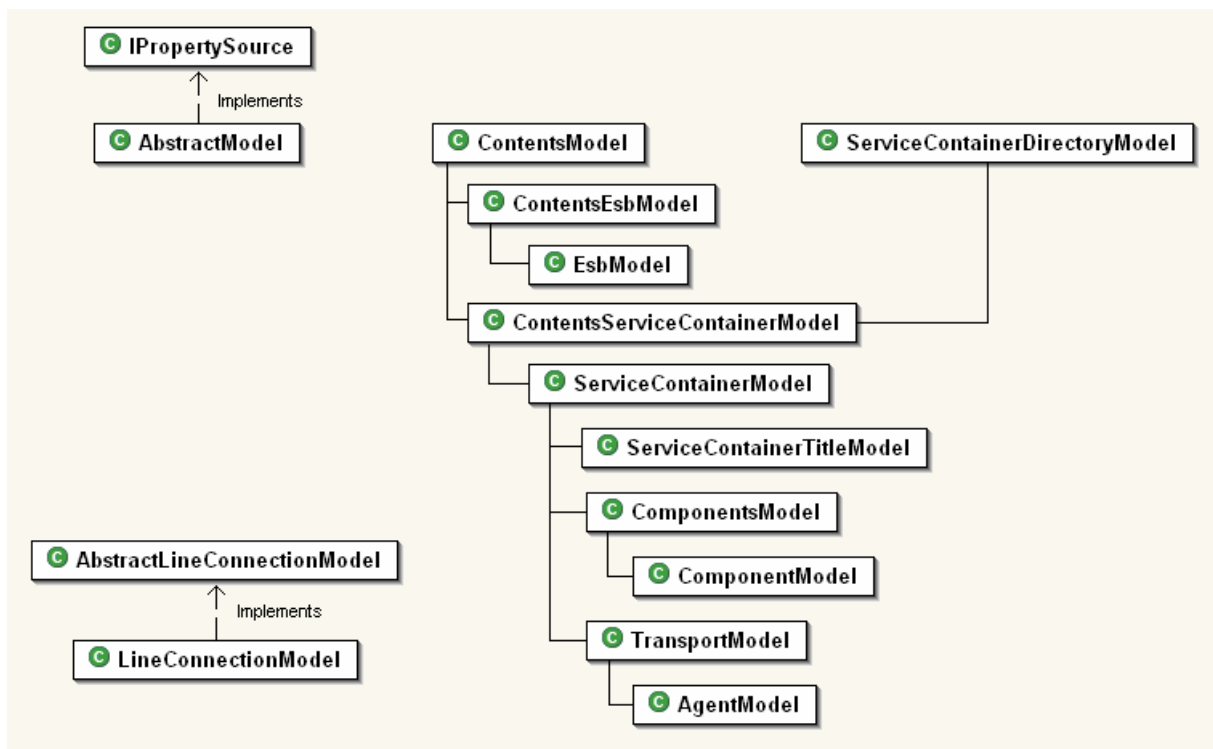


Figure 8 : Models

org.objectweb.petals.admin.actions

This package contains all needed classes for action on components, like start,stopshutdown,etc... on a System service for example

Each action have a state. Each state can have several actions whereas an action belongs to one state.

All actions are automatically built via states.xml file, which defined what to do for each actions and their states

org.objectweb.petals.admin.adapter

The classes in this package is the bridge between graphical actions and real actions using JMX api.

org.objectweb.petals.admin.editparts

Editpart are controllers of your models.

See GEF documentation for further information

org.objectweb.petals.admin.editpolicies

Editpolicies allow you to react (or not) to user interactions with your models.

See GEF documentation for further information

org.objectweb.petals.admin.models.commands

Commands are a simple way to modify your models in response to user actions on screen

See GEF documentation for further information

org.objectweb.petals.admin.models.states

For now, this package contains the super class of all states

org.objectweb.petals.admin.models.tools

This package is made of various tools needed to build these plug-in, like custom figures or model builders.