



## PDE Declarative Services – Graduation Review

Rafael Oliveira Nóbrega  
Chris Aniszczyk



## Outline

- Overview
- Active Communities
- Open Source Operation
- IP Log
- Future



## Overview

- Declarative Services is part of the OSGi Compendium and provides a lightweight way to declaratively wire services together
- The PDE DS work was going on in the PDE Incubator as part of the Google Summer of Code (GSOC) program
- The PDE DS work now provides
  - A Model for DS Files
  - UI Editor
  - Code Assistance
  - Basic Error Reporting and Validation



## What is graduated and where

- Three PDE DS plug-ins to be graduated to PDE:
  - `org.eclipse.pde.ds.core`
  - `org.eclipse.pde.ds.ui`
  - `org.eclipse.pde.ds.tests`
- A new group `pde-ds` will be created
- A new bugzilla component 'Declarative Services'



## Active Communities

- Developer Community
  - PDE UI Team
    - Chris Aniszczyk - Code 9
    - Curtis Windatt - IBM
  - Student
    - Rafael Oliveira Nóbrega – GSoC 2008 Student Developer



## Open Source Operation

- The DS development has been ongoing in the PDE Incubator component of the Eclipse Project Incubator
  - Use Bugzilla (bugs.eclipse.org) for workflow using the “Eclipse – PDE - Incubator” product and component.
  - Use pde-dev mailing lists, IRC and public phone call for developer discussions
  - Maintain a wiki
    - <http://wiki.eclipse.org/PDE/Incubator/DS>



## IP Log

- The developers understand and adhere to the Eclipse Development Process, committer responsibilities and due diligence rules, as well as the Eclipse IP Policy.
- We use the Automated IP Log
  - [http://www.eclipse.org/projects/ip\\_log.php?projectid=eclipse.incubator.pde](http://www.eclipse.org/projects/ip_log.php?projectid=eclipse.incubator.pde)



# Q & A

- Any questions!?

The screenshot displays the Eclipse IDE's editor for a service.xml file. The interface is split into two main sections: 'Content' and 'Definition'.

**Content View:** Shows a tree structure for the 'service' component. Under 'service', there is a package 'org.mytest.class' and a reference 'reference1'. To the right of the tree are several buttons: 'Add Service Component', 'Add Property', 'Add Reference Service', 'Add Properties', and 'Add Provided Service'.

**Definition View:** Titled 'Definition', it contains the text 'Specify the service's component attributes:'. It features four input fields:

- Name\*:** A text box containing the value 'service'.
- Factory:** An empty text box with a 'Browse...' button to its right.
- Enabled:** A dropdown menu currently set to 'true'.
- Immediate:** A dropdown menu currently set to 'false'.