Development with the INGENIAS Development Kit

Instructions for working at the same time with Eclipse and the IDK

Jorge J. Gómez Sanz

UCM UNIVERSIDAD COMPLUTENSE MADRID

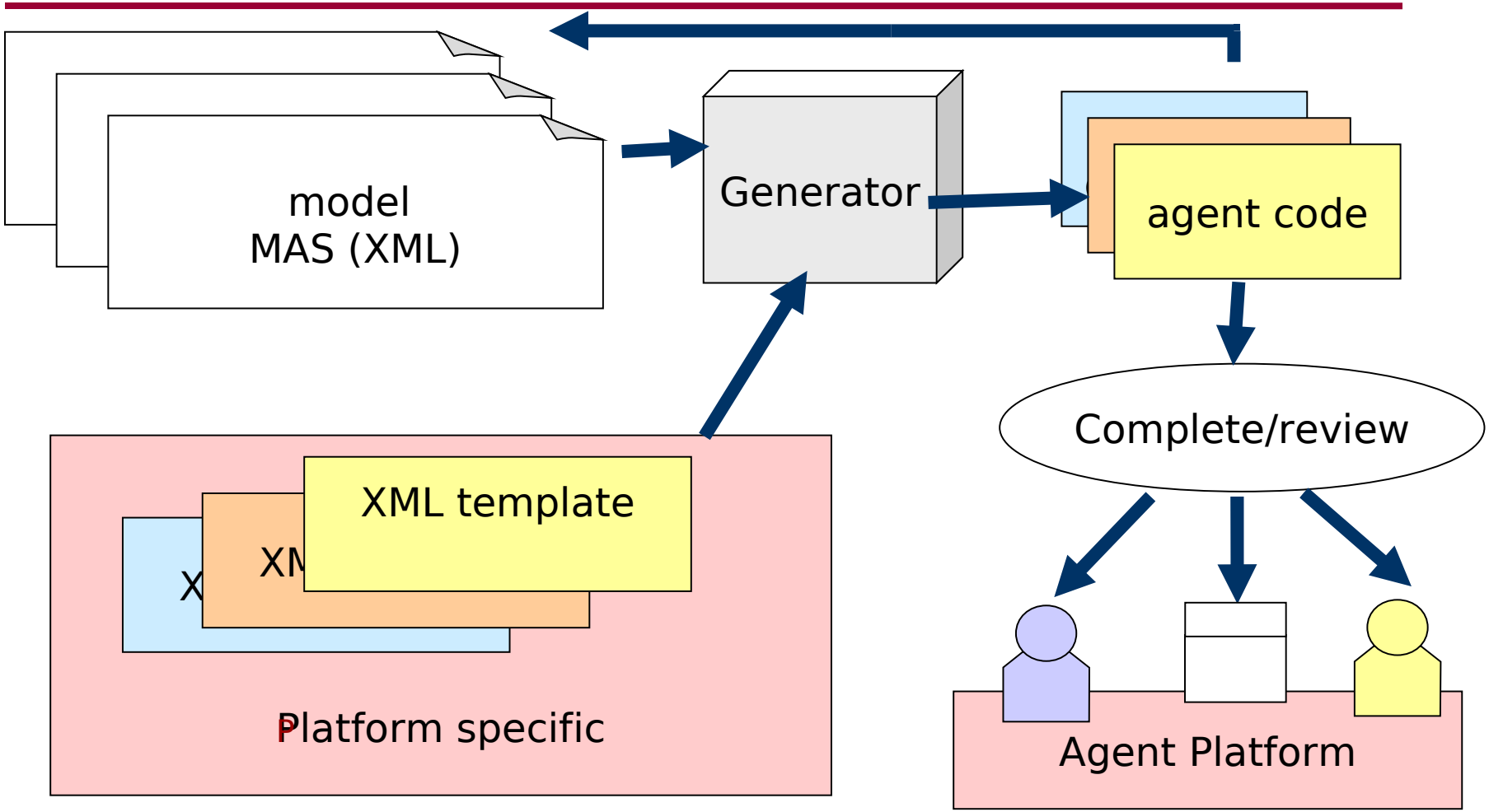
Dep. de Ingeniería del Software e Inteligencia Artificial

http://grasia.fdi.ucm.es

Agent development with IDK

- IDK provide support for INGENIAS methodology
  - Methodology: notation + tools + development process + paradigm

IDK:Code generation scheme



About the IDK

- It is based on a lenguaje for MAS modeling
  - The language is expressed as a meta-model
- The language is a superset of what the current code generators recognise
  - Not all the elements of the specification are used actually
- Many different code generators can be built
  - According to different interpretations of what the modelling language means

About the IDK

- It is developed with Java
  - Compatible with versions superior to 1.4
- It uses ant tool
  - http://ant.apache.org
- Uses GPL software
  - Libraries for graphics management
  - Libraries for XML persistence
  - JADE Platform

About the IAF

- It is the INGENIAS Agent Framework
  - The main code generator for the IDK
- It produces pure Java code working over the JADE Platform. It includes facilities for
  - Mental state management
  - Coordination
  - Decision taking
  - Debugging & Deploying & Testing
- The MAS generated with the IAF are almost fully developed
  - Task code is missing
  - Coordination, control flow, and deployment is already implemented
- A MAS usually requires many non-agent software which is not produced by the IAF

Downloading the software (I)

Browse **http://ingenias.fdi.ucm.es/hudson/job/INGENIASDevelopmentKitBuilds/**

- Download **ingenias-all-XXX.zip**
- Unzip and in folder **foo**
- Launch a console with access to ant and java binaries
  - Execute **ant runide**

Downloading the software (II)

Browse **http://ingenias.fdi.ucm.es/hudson/job/INGENIASDevelopmentKitBuilds/**

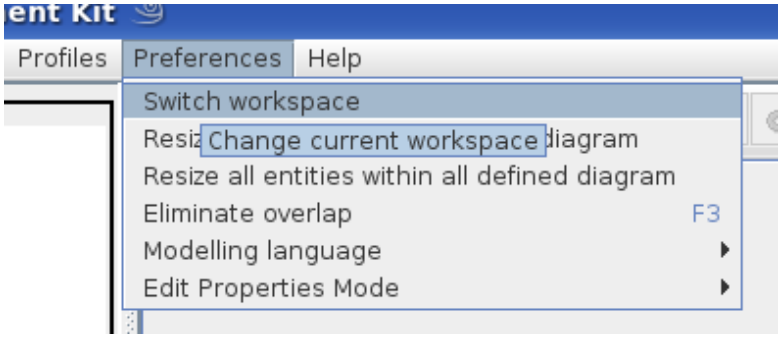
- Download **ingenias-installer-XXX.jar** and then
  - Execute the jar file from command line
    - Java -jar ingenias-installer-XXX.jar
  - Or double click from your file browser on the file
- Follow the instructions on the emerging dialogue to proceed with the install
- New desktop shortcuts as well as entries in your desktop menu will be created to launch the editor
- Launch the IDK

Steps of the tutorial

1) Set the workspace folder
2) Create a project using a hello world template
3) Execute the resulting system from console
   1) Mode prod stand alone
   2) Jade combined mode
4) Modifying the generated project from Eclipse

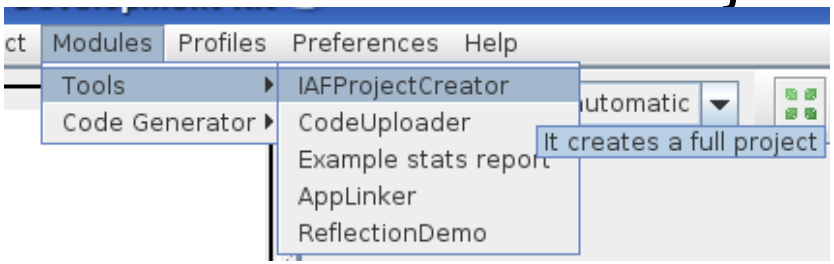
Setting the workspace

- Select a folder in your PC to act as workspace folder
- Select the switch workspace option from the preferences menu and either type in the path to that folder or browse using the file dialog

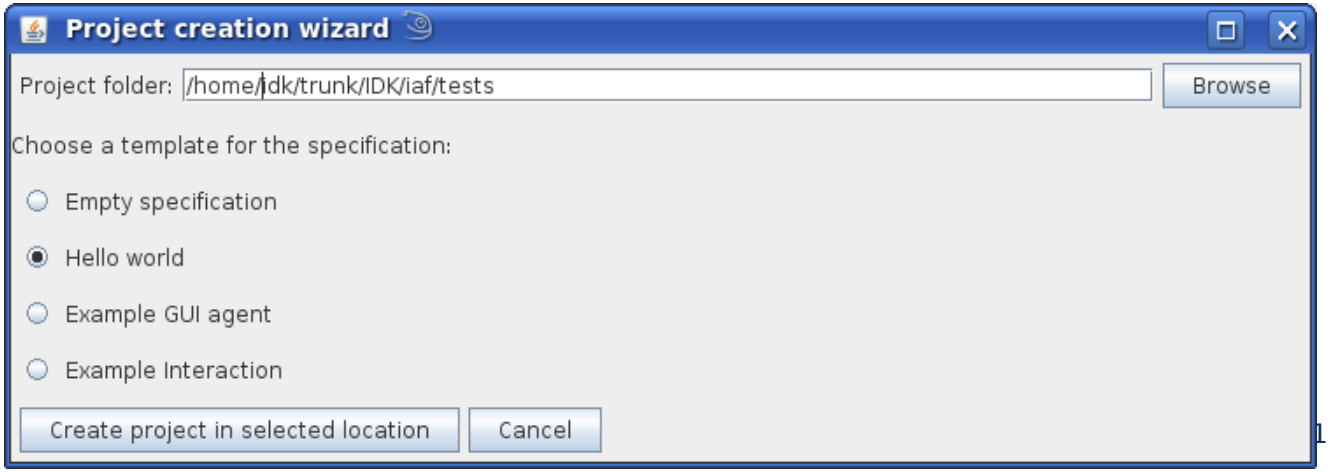


Creating a project (I)

- In the editor, go to the Modules section, then tools, and then to the IAFProjectCreator entry



- As a result, the dialog shown below will appear. Type the path to the project folder (it may or may not be created) and choose the create button

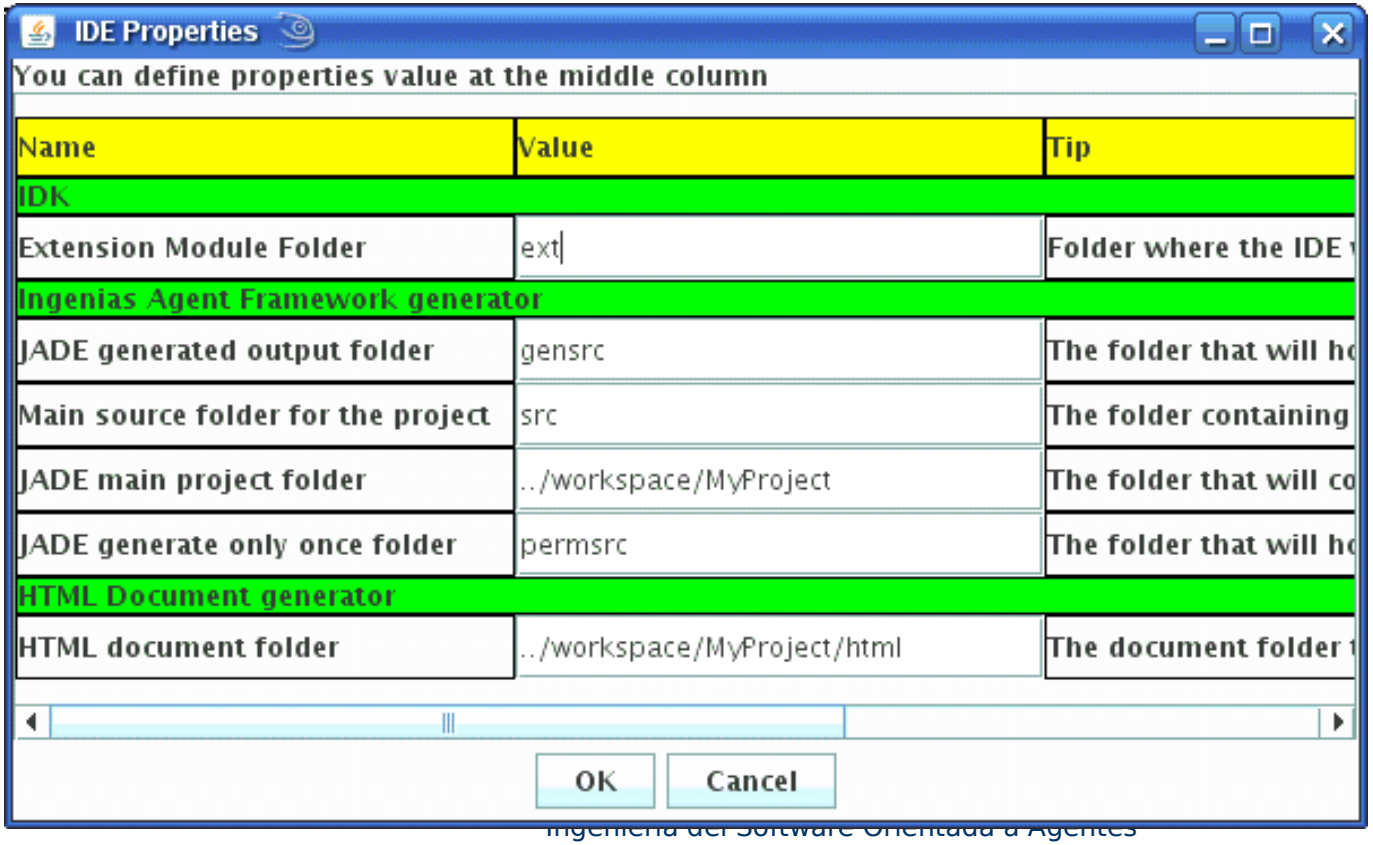


Creating a project (II)

- The creation itself will take a little since the example specification and different libraries have to be copied to the desired folder, plus the generation of the MAS has to take place. Hence, the progress dialog will seem to be stuck at 46%, but it will continue after a little.

Creating a project (III)

- Once created, it is worth inspecting the paths associated to the specification.
- Go to menu Project -> Properties

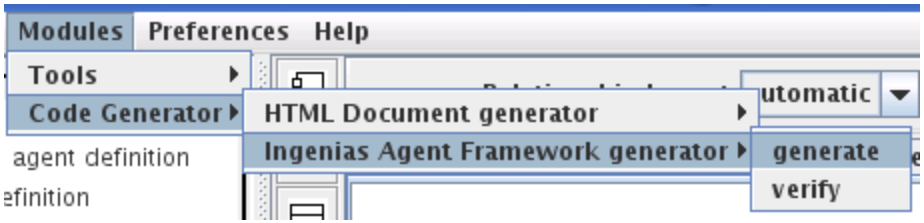


Creating a project (IV)

- The properties you should pay attention to are:
  - HTML Document folder. Its value should be
    - {workspace}/myprojectname/html
  - JADE Main project folder. Its value should be
    - {workspace}/myprojectname
- If the values do not correspond to the ones above, something may have failed during the project creation stage

Creating a project (V)

- A correct properties configuration permits to generate code in the appropriate locations
- Go to the menu option Modules -> Code Generator -> Ingenias Agent Framework generator -> Generate
- Logs window section will present the result of the code generation



Why integrating with Eclipse

- Eclipse is a major development platform for different programming languages
- Most of existing agent oriented support tools are migrating to this platform
- Migration of INGENIAS towards Eclipse is being handled at various levels
  - Meta-model. A EMF version is being tested to replace the old XML based persistence
  - Project management. A plugin for the actual project definition and maintenance
- The IAF plays an important role since it produces the projects to be managed with Eclipse
  - Until the plugin is ready, some tasks have to be performed manually

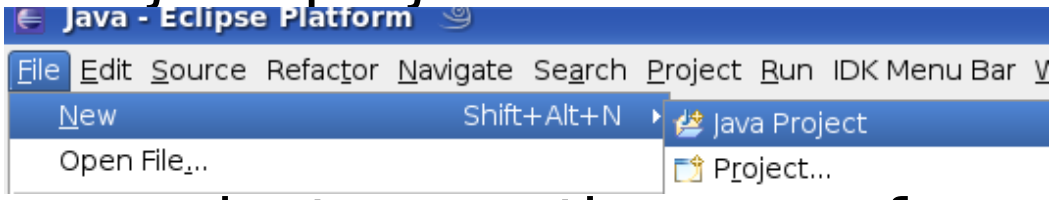
Creating a project within Eclipse

- Creating a project in Eclipse with content from the recently created project can be handled in different ways.
- First, let us consider the different involved folders
  - Your created project folder (**foo** folder) is in
    - XXXX
  - The eclipse workspace is in
    - YYYY
- Cases to consider

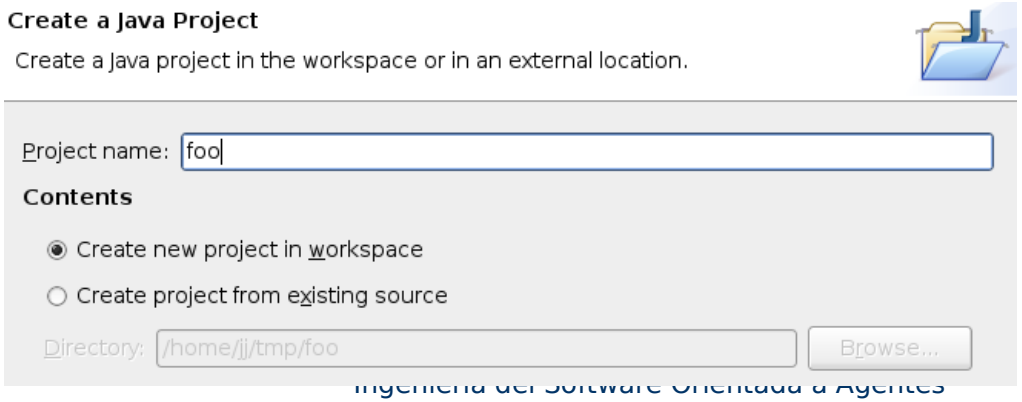
  1) XXXX=YYYY
  2) XXXX!=YYYY

Case 1: XXXX=YYYY

- In this case, the workspace of the IDK and workspace from Eclipse are the same
- The name of the project in the IDK matches the name of its containing folder, **foo** in this case
- Create new java project

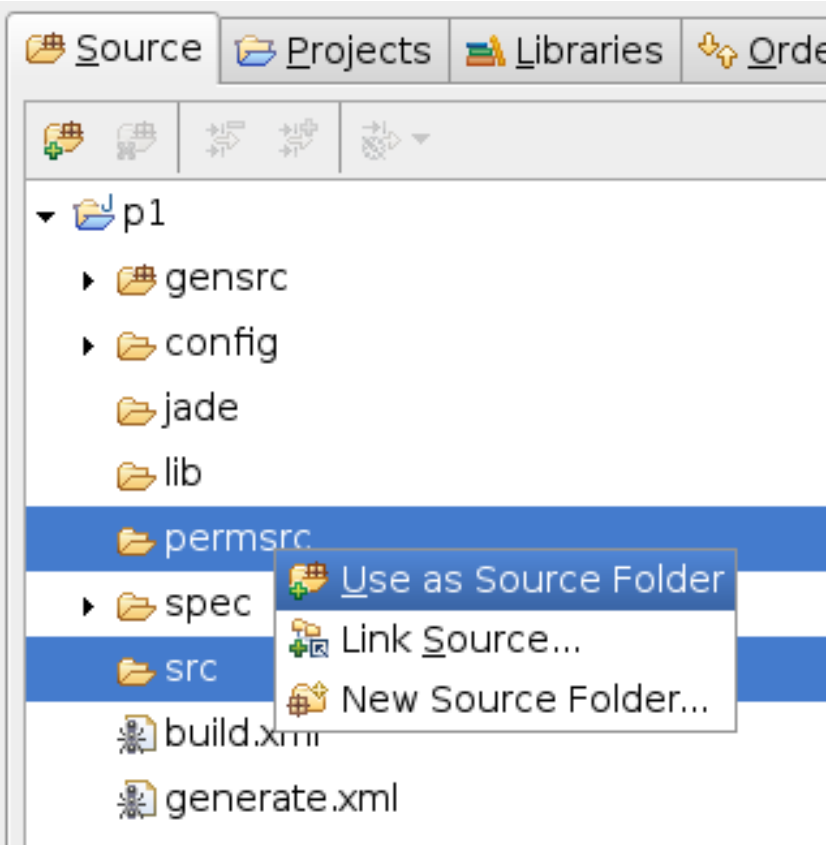


- Choose as project name the name foo and click next



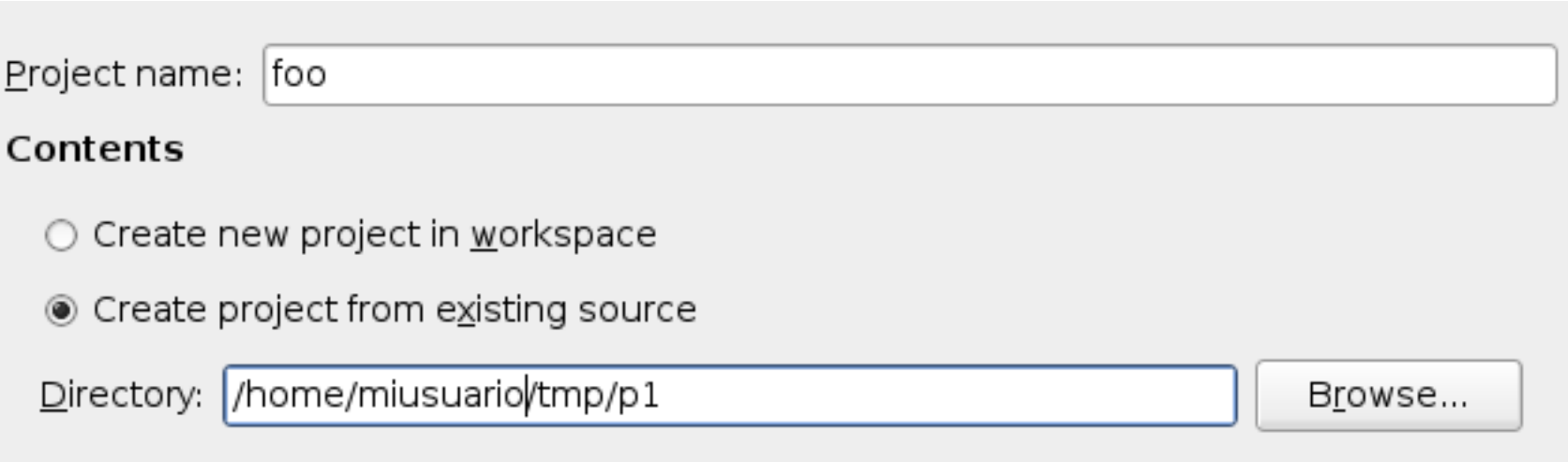
Case 1:XXXX=YYYY

- Choose **src**, **gensrc**, and **permsrc** as source folders. To do so, select those folders as shown below, right click and choose **use as source folders**
- Now, press finish



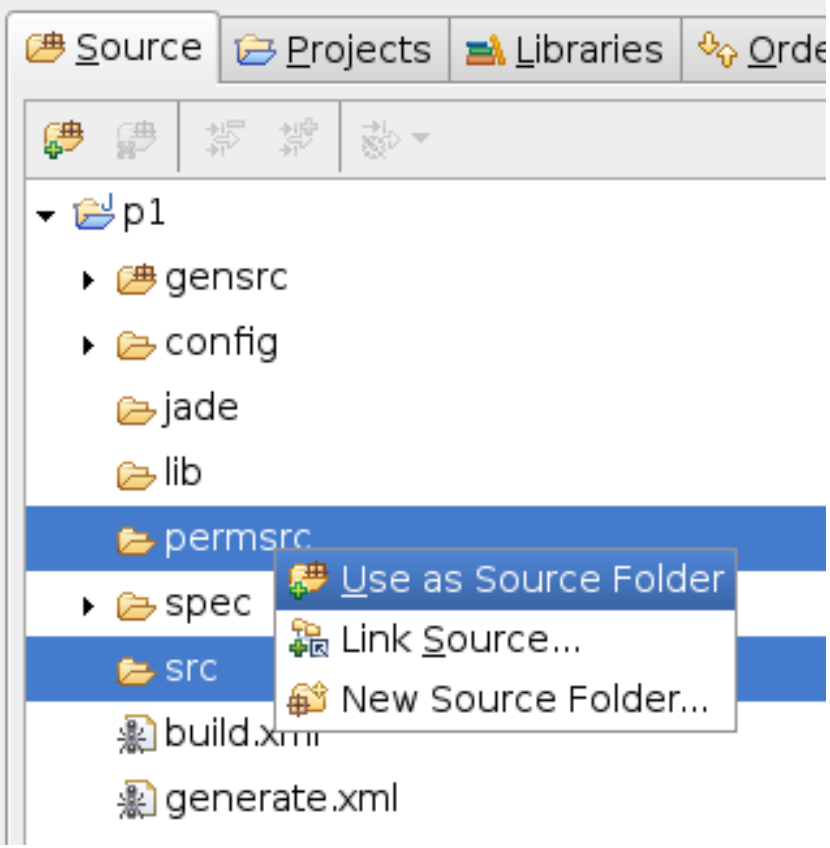
Case 2: XXXX!=YYYY

- In this case, do proceed the same as before, i.e., try to create a new java project
- This time, choose **create project from existing source**, and write down the path to the IDK project, then press **next**



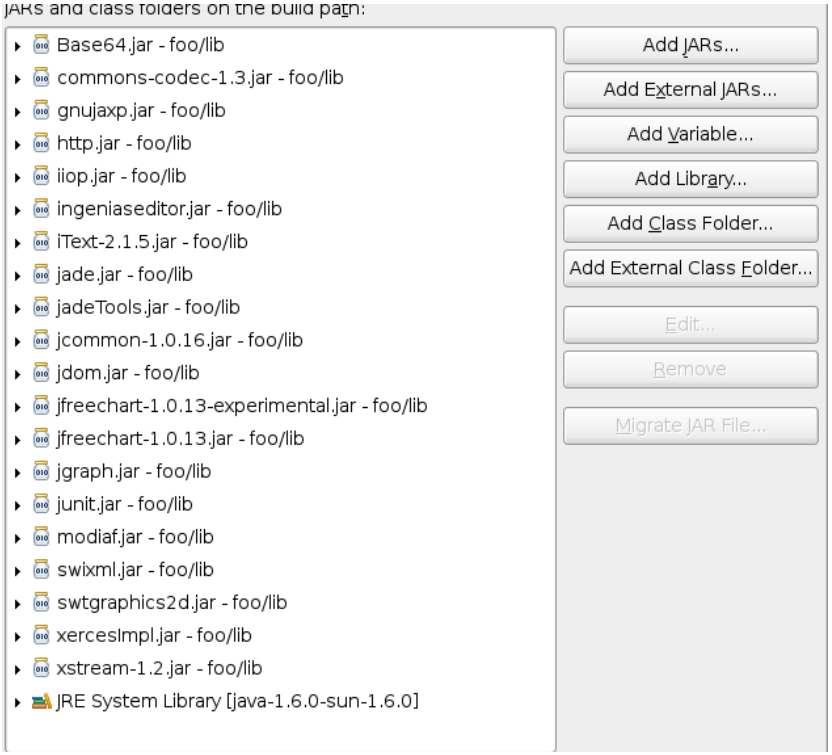
Case 2: XXXX!=YYYY

- Again, select the necessary sources
- Choose **src**, **gensrc**, and **permsrc** as source folders. To do so, select those folders as shown below, right click and choose **use as source folders**



Case 2: XXXX!=YYYY

- You may need to configure libraries.
- If your project creation wizard does not look like the right image, try to fix it by choosing the same jars from the lib folder of your IDK project

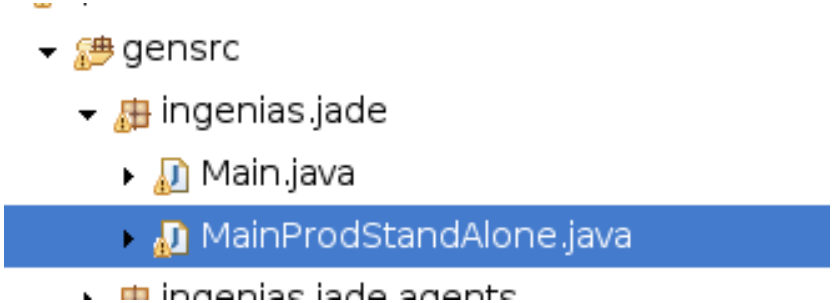


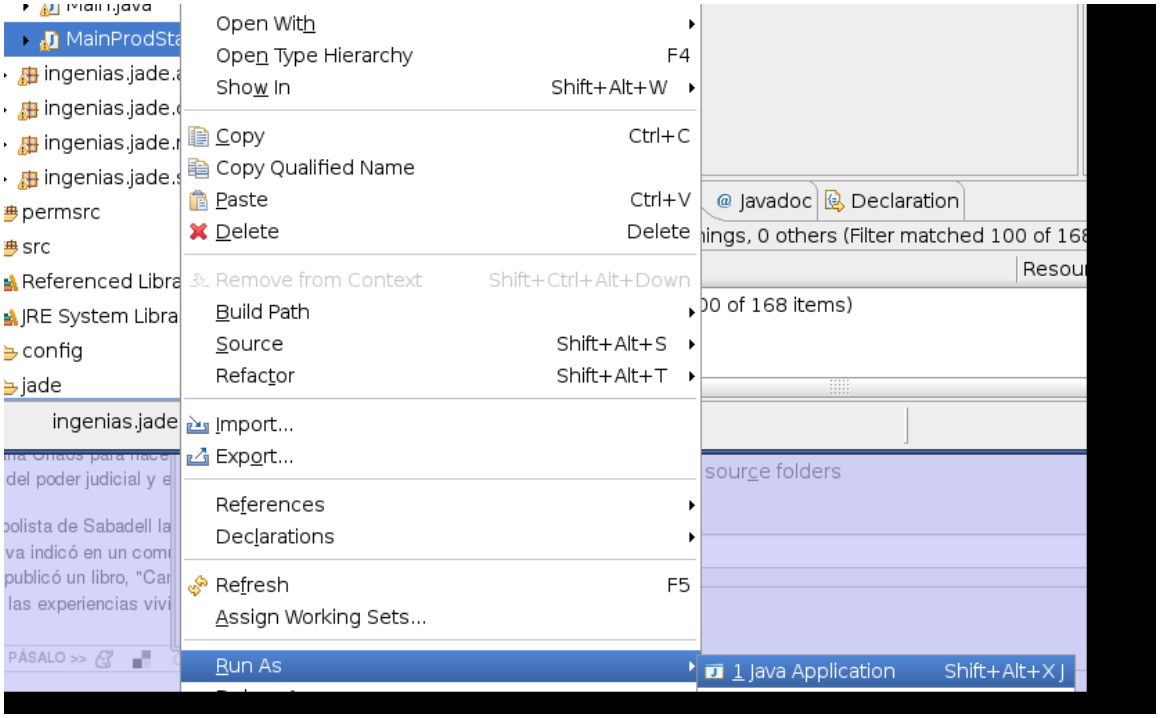
Executing from Eclipse

- There are two ways of executing a MAS from Eclipse
  - Using the same main java files
  - Through ant

Using the same java files

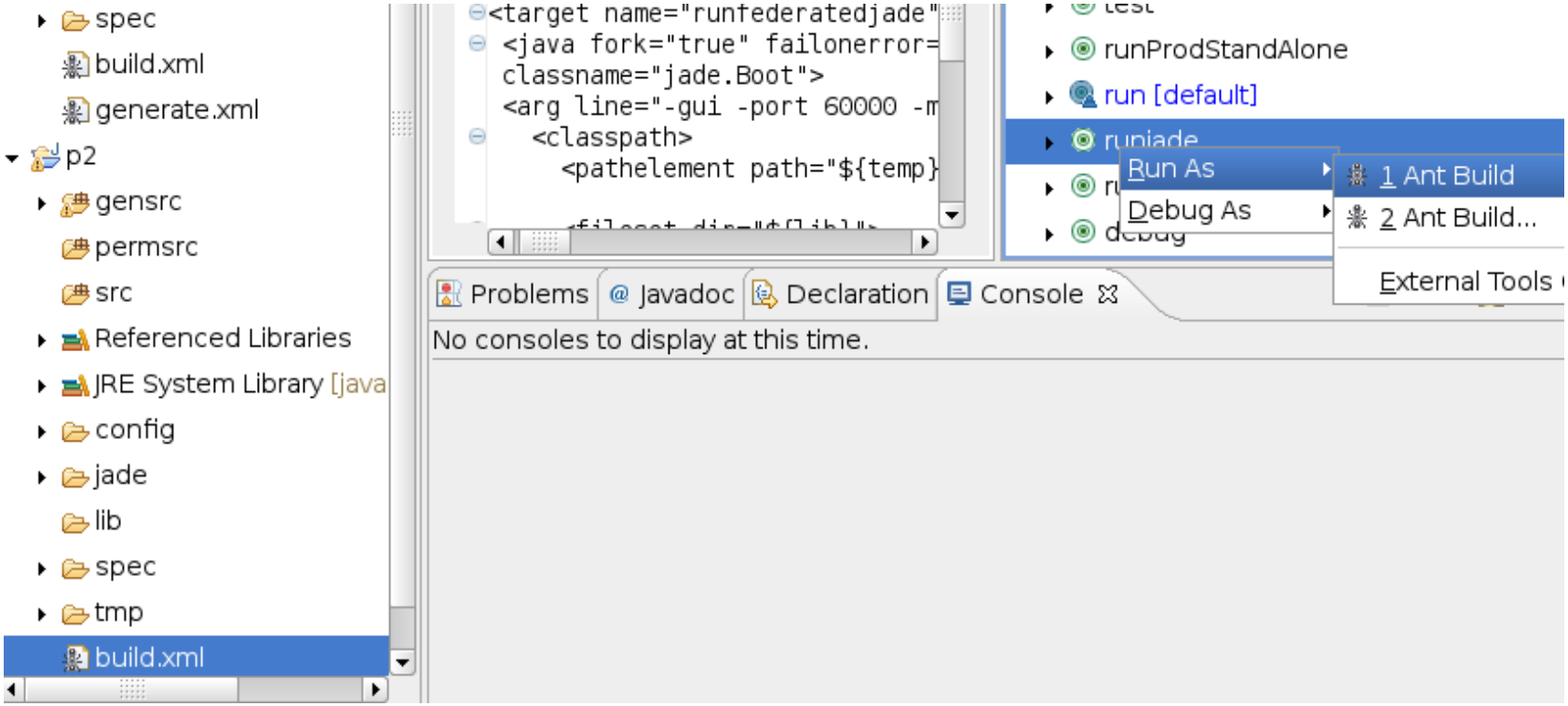
- Go to your package explorer and open the gensrc folder.
- Unfold until you see several files with the prefix **Main**
- Choose one, right click on it and choose **run as java application**
- This method will work with all main files with suffix **ProdStandAlone**





Using the ant file

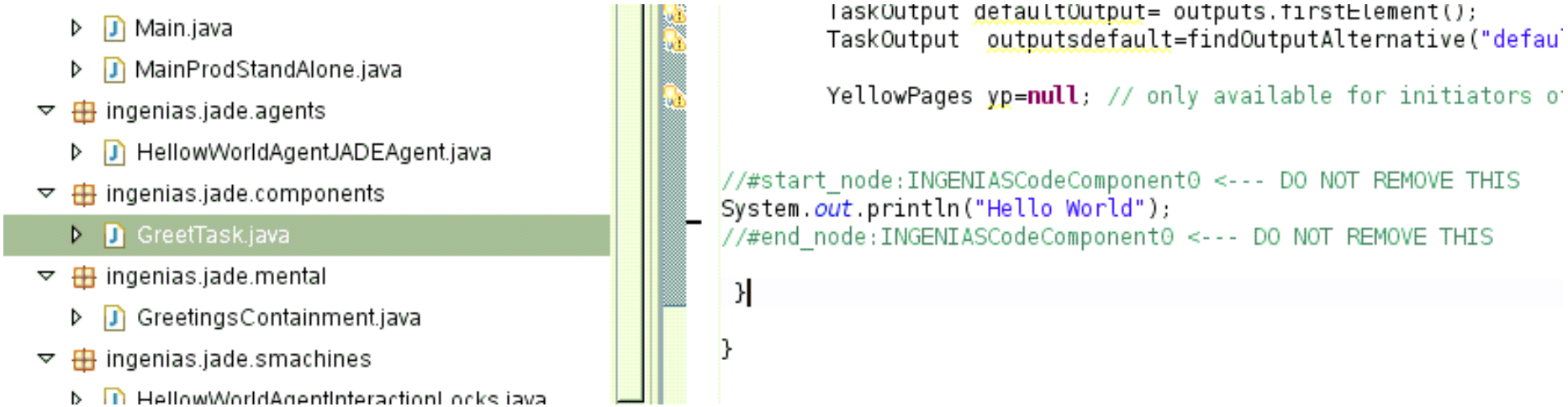
- Check your outline view is open.
  - If it is not, go to window -> show view -> outline
- Select in the package explorer the build.xml file produced by the IDK
- Select in the outline view the appropriate target and then right click to execute the target



To launch the produced MAS

- There are two options
  - Conf 1. You launch only prod stand alone targets. These can be launched from ant or directly executing those **gensrc/ingenias/jade/MainXXXProdStandAlone.java** classes
  - Conf 2. Using the ant launching option, you launch jade (target runjade in the build.xml).
    - The MAS is launched either by selecting run targets in the build.xml file(targets with prefix run and without suffix ProdStandAlone)
    - Or launched executing files **gensrc/ingenias/jade/MainXXX.java** without ProdStandAlone suffix.
- Conf 2 provides additional GUIs to debug the MAS, concrete the JADE management GUI and a debugging GUI for INGENIAS agents

Modifying a task

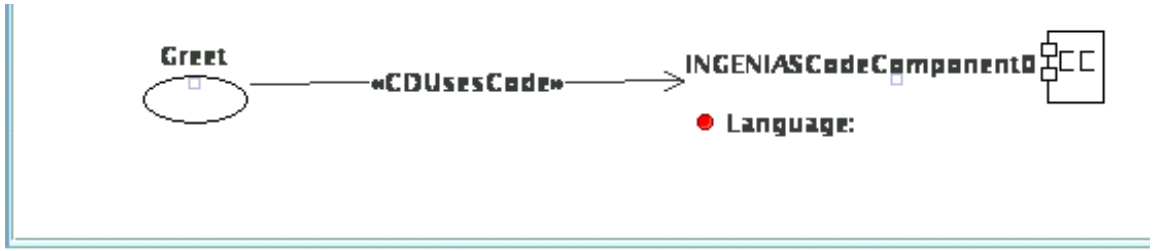


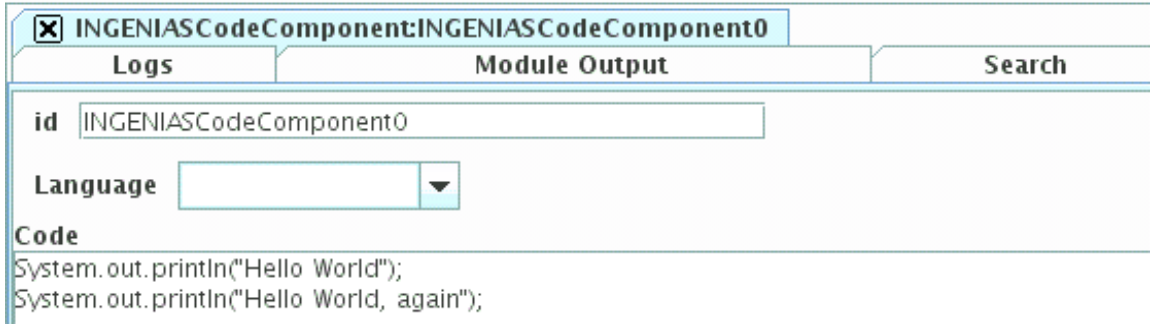
- Tasks are generated inside **gensrc/ingenias/jade/components**
  - Look for the name of the task as it appears in the specification
- Modifications have to be copied to the editor
  - Look at the experimental features in the training page for other alternatives

Modifying a task

- In the example, the task is modified to incorporate a new hello world statement
- The new statement has to be copied in the specification to keep this change when the code is regenerated

```
//#start_node:INGENIASCodeComponent0 <--- DO NOT REMOVE THIS
System.out.println("Hello World");
System.out.println("Hello World, again");
//#end_node:INGENIASCodeComponent0 <--- DO NOT REMOVE THIS
```





Warning

- Changes different from adding code to the assigned location in the task could be removed in the next code generation
  - For instance, a developer cannot add a new method to the task
    - Instead, create these new methods in external classes and access them from the task either through public static implementations or through a singleton pattern
- Beware of including new imports in the task Java class. These imports will be removed in the next code generation
  - Use fully qualified names

Eclipse Preferences

- To avoid the use of imports, and refresh automatically the project, the eclipse preferences have to be set as indicated to the right
- You can open preferences with the menú Window->Preferences