Detailed Conception of Econet plugin

In this section, the different stages of the Econet plug-in conception will be explained. Firstly, the rule-based system will be explained , entities used in the process, then UML conception that concluded with the realization of the first version of itself;

Rules System

Document : specification des règles

Econet plugin structure

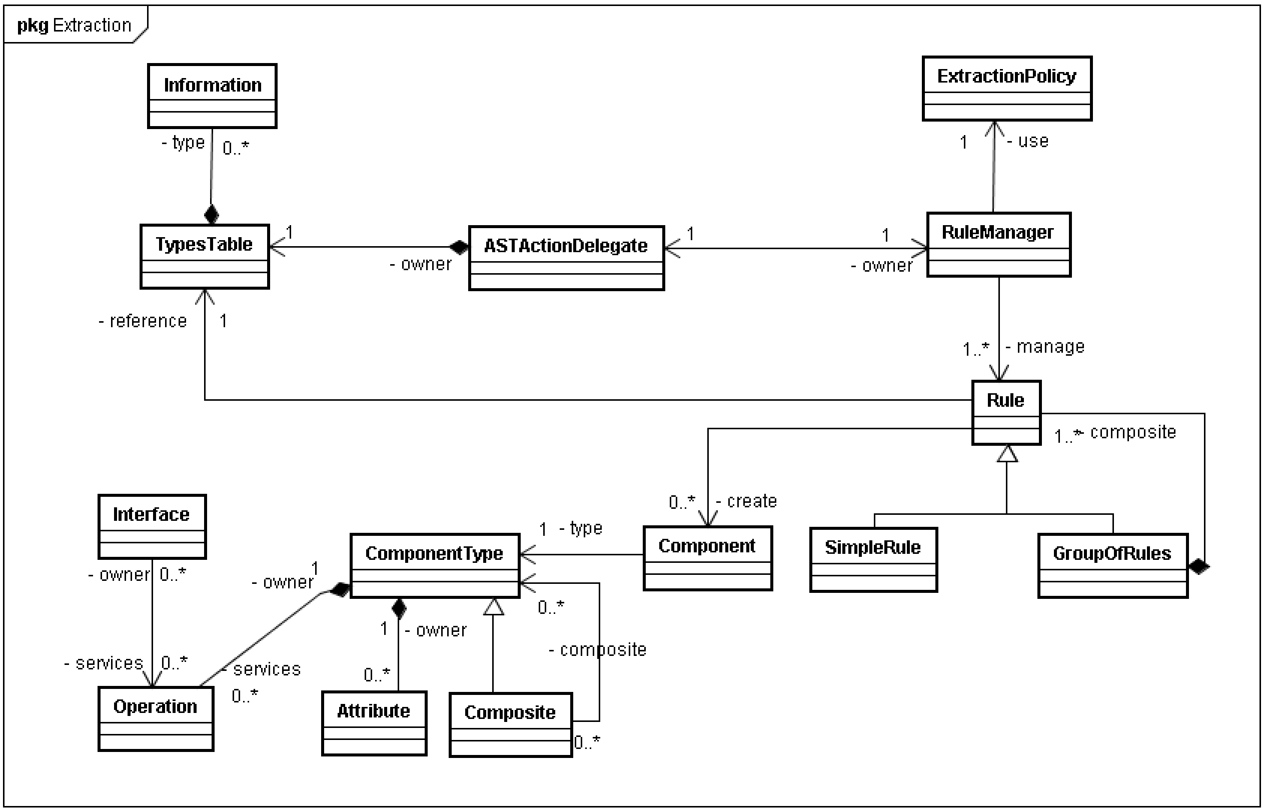
So far, the system's rules have been defined. However, entities necessary for the extraction process of a Java component structure have to be defined.

After reflection, the extraction program has been broken down in a number of classes.

One needs:

- A main class that instructs start command to the others;it initially collects the project whose component structure is to be extracted and calls upon the rules manager to apply in order rules provided by the user. This class will contain information regarding data types of Java code and an AST whose structure will be used for the extraction rules
- A class that will be play the rules manager, applying rules chosen by the user and provided by extraction policy on the process. These rules are of type simple and/or complex.
- A policy rules extracting class that collects rules to be applied during process in the rules manager.
- A component class that will model a component, its type and structure.

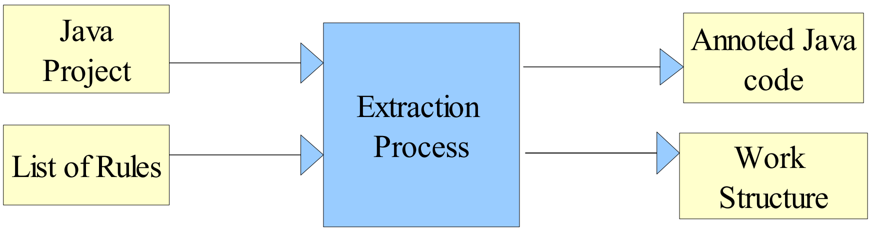
The following class diagram was then established:



Class names were chosen as follows:

- ***ASTDelegate*** : The main class.
- ***TypesTables*** : Class containing persistent data names and information.
- ***Information*** : Class detailing information about a type of persistent data structure
- ***RuleManager*** : Class acting as the rules manager
- ***ExtractionPolicy*** : Class that collects in a file the order of the rules as defined by the user.
- ***Rule*** : Abstract class defining extraction rule.
- ***SimpleRule*** : A ***Rule*** subclass representing a simple rule. .
- ***GroupRule***: A ***Rule*** subclass representing a set of rules simple and/or complex.
- ***Component*** : Class defining a component associated with a component type ***ComponentType***, that holds attributes ***Attribute***, operations ***Operation*** interfaces ***interfaces*** and whose structure could be composite ***Composite***.

The following figure illustrates the ins and outs of the extraction program.



The program must:

- Take as an argument the name of a Java project existing in the workspace environment and a file containing order of the rules to apply in the process chosen by the user.
- Supply an annotated Java code as well as a work structure in return. The work structure will contain information about the extraction process on rules applied throughout the process, their results, temporary elements (classes, types, interfaces, methods) with their appearance level (stage in the process) and verification of the structure result.

Our renamed Econet project, is a plug-in intended to work on Eclipse. It will therefore be implemented in Java, and will comply to inherent construction norms of plug-ins.

Eclipse

Eclipse is an IDE: Integrated Development Environment. Eclipse is the project principal tool and is a very powerful development platform.

Eclipse kernel is actually composed of :

- A base capable of loading modules (plug-ins)
- Integrated modules enabling management of a set of resources (projects, files, folders,...)
- Modules enabling creation of coherent graphical interfaces. Using this library, plug-ins keeps a homogeneous aspect.

The Java development part is indeed a set of plug-ins that constitute the first usage of the base. Their principal characteristic is to be delivered with Eclipse. Note that plug-ins forming C++ development environment are also available, although not in Eclipse standard version, it requires a separate download.

Eclipse could therefore be used as the basis of any given development tool, no matter the language and the file formats supported.

Eclipse is found in different guides, workbench user guide, Java development user guide, Platform plug-in developer guide, JDT plug-in developer guide, PDE guide, Eclipse UML plugin user guide.

Plugin integration in Eclipse environment

(soon)