The Micro-structures detector

A component of the MARPLE project

We have developed a plug-in for the Eclipse platform [3], called *Micro-structures detector*, which is part of the MARPLE project [1]. This module is devoted to the identification of elemental design patterns [6], design pattern clues [7, 8] and micro patterns [5] inside subject systems. Figure 1 depicts the plug-in architecture.

The Micro-structures detector is laid on the functionalities provided by the Eclipse framework. In order to be analyzed, the source code of the subject system needs to be translated into an Abstract Syntax Tree (AST) representation. The tree data structure is inspected by visitors [4], which have the aim to detect realizations of micro-structures inside the system. For each single micro-structure, a visitor has been implemented. The AST representation and the basic classes to implement the visitor functionalities are provided by the Eclipse Java DOM/AST library, which contains those classes that model the source code of a Java program as a structured document.

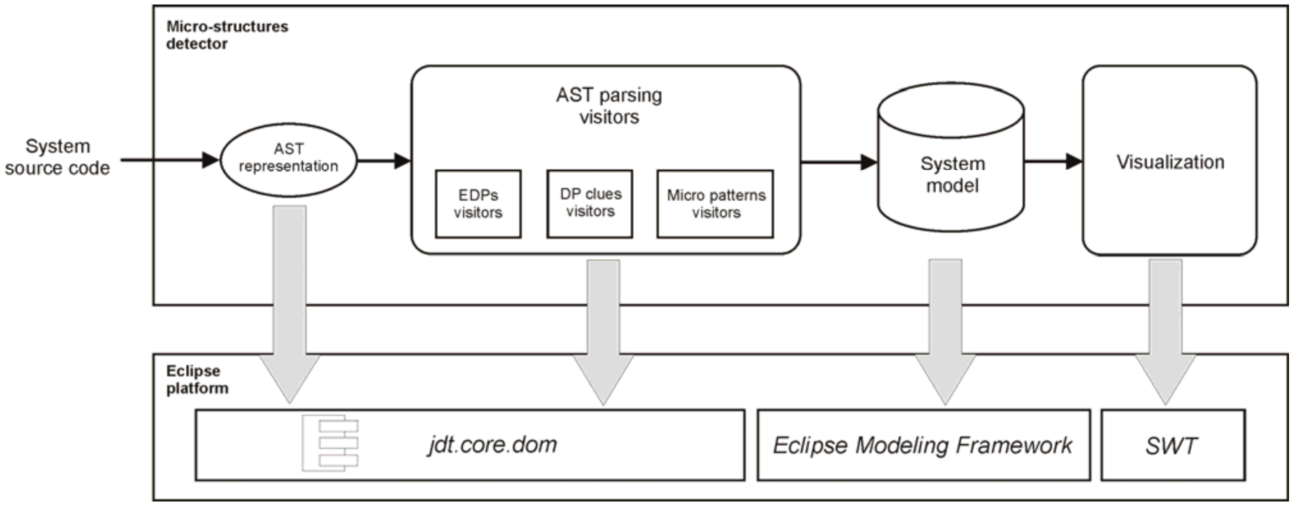
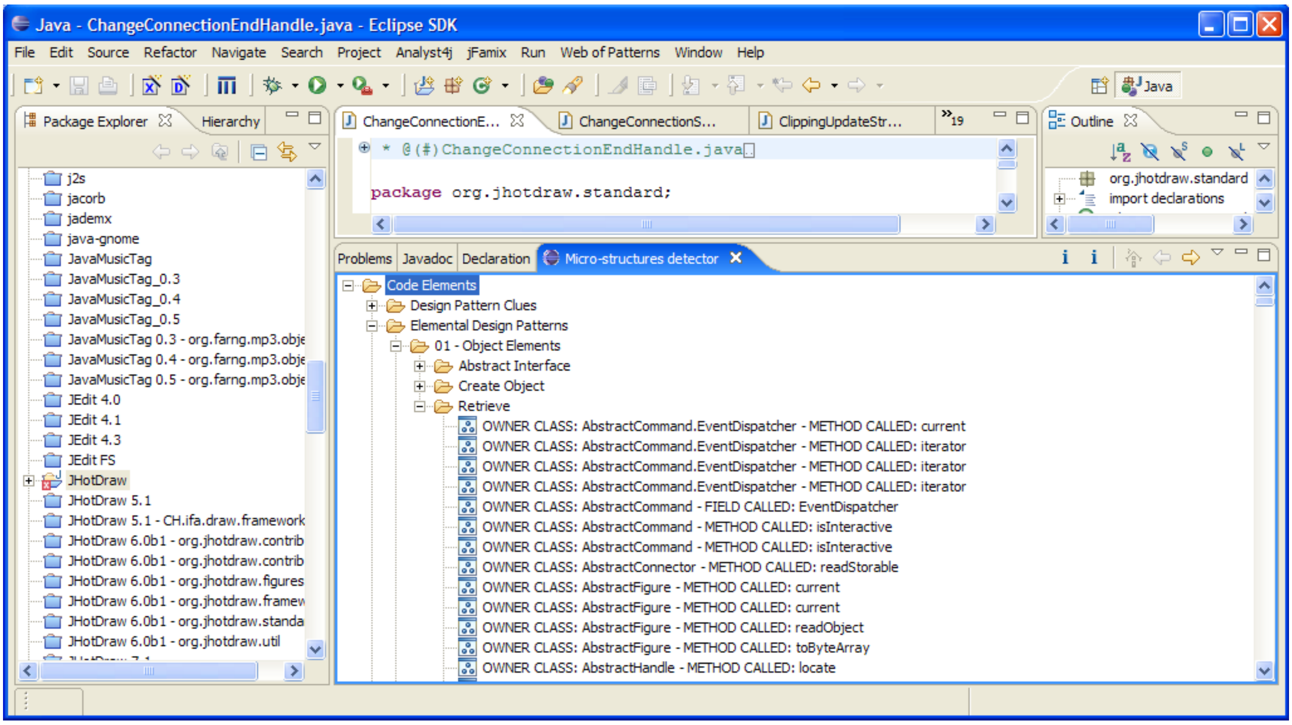


Figure 1 *– The architecture of the Micro-Structures Detector*

The micro-structures instances detected by the visitors are then stored in a model [2], built on top of the Eclipse Modeling Framework (EMF), which organically represents all the system classes and interfaces, reporting for each of them the micro-structures that have been detected within it. The model is generated in order to allow an easy recovery of the stored information that is to be used for the DPD and SAR activities.

Finally, the micro-structures instances can be shown to the user. The visualization module is laid on the Eclipse Standard Widget Toolkit (SWT), and basically provides a comfortable tree view in which all the micro-structures instances are collected according to their categories. Figure 2 reports a screenshot of the Micro-structures detector output obtained on the analysis of JHotDraw 6.0b1. In the figure we can notice the results structure, which are organized in folders according to a tree structure collecting the considered micro-structures categories and each single micro-structure. In this example, we can see a subset of the detected Retrieve EDP instances.



Java - ChangeConnectionEndHandle.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Analyst4j jFamix Run Web of Patterns Window Help
Java
Package Explorer
Hierarchy
ChangeConnectionE...
ChangeConnectionS...
ClippingUpdateStr...
»19
Outline
org.jhotdraw.standard
import declarations
j2s
jacorb
jademx
java-gnome
JavaMusicTag
JavaMusicTag_0.3
JavaMusicTag_0.4
JavaMusicTag_0.5
JavaMusicTag 0.3 - org.farng.mp3.obje
JavaMusicTag 0.4 - org.farng.mp3.obje
JavaMusicTag 0.5 - org.farng.mp3.obje
JEdit 4.0
JEdit 4.1
JEdit 4.3
JEdit FS
JHotDraw
JHotDraw 5.1
JHotDraw 5.1 - CH.ifa.draw.framework
JHotDraw 6.0b1 - org.jhotdraw.contrib
JHotDraw 6.0b1 - org.jhotdraw.contrib
JHotDraw 6.0b1 - org.jhotdraw.figures
JHotDraw 6.0b1 - org.jhotdraw.framew
JHotDraw 6.0b1 - org.jhotdraw.standa
JHotDraw 6.0b1 - org.jhotdraw.util
Problems
Javadoc
Declaration
Micro-structures detector

```
* @(#)ChangeConnectionEndHandle.java

package org.jhotdraw.standard;
```

```
Code Elements
  Design Pattern Clues
  Elemental Design Patterns
    01 - Object Elements
      Abstract Interface
      Create Object
      Retrieve
        OWNER CLASS: AbstractCommand.EventDispatcher - METHOD CALLED: current
        OWNER CLASS: AbstractCommand.EventDispatcher - METHOD CALLED: iterator
        OWNER CLASS: AbstractCommand.EventDispatcher - METHOD CALLED: iterator
        OWNER CLASS: AbstractCommand.EventDispatcher - METHOD CALLED: iterator
        OWNER CLASS: AbstractCommand - FIELD CALLED: EventDispatcher
        OWNER CLASS: AbstractCommand - METHOD CALLED: isInteractive
        OWNER CLASS: AbstractCommand - METHOD CALLED: isInteractive
        OWNER CLASS: AbstractConnector - METHOD CALLED: readStorable
        OWNER CLASS: AbstractFigure - METHOD CALLED: current
        OWNER CLASS: AbstractFigure - METHOD CALLED: current
        OWNER CLASS: AbstractFigure - METHOD CALLED: readObject
        OWNER CLASS: AbstractFigure - METHOD CALLED: toByteArray
        OWNER CLASS: AbstractHandle - METHOD CALLED: locate
```

Figure 2 *– A sample of the results provided by the Micro-structures detector on the analysis of JHotDraw 6.0b1*

The Micro Structure Detector is available on request: marco.zanoni@disco.unimib.it

References

[1] F. Arcelli, C. Tosi, M. Zanoni, and S. Maggioni, A Tool for Design Pattern Detection and Software Architecture Reconstruction. In Information Sciences 181 (2011) 1306–1324, Elsevier, 2011.

[2] F. Arcelli, C. Tosi, M. Zanoni, R. Porrini, M. Vivanti, S. Maggioni, A Model Driven Approach for Program Comprehension to Support Software Evolution, Technical Report, Dipartimento di Informatica, Sistemistica e Comunicazione (DISCo)-20-03-09, University of Milano-Bicocca, 2009.

[3] Eclipse framework, http://www.eclipse.org/

[4] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison Wesley, 1994.

[5] Y. Gil, I. Maman, Micro Patterns in Java Code, in Proceedings of the 20th annual ACM SIGPLAN conference on Object oriented programming, systems, languages, and applications (OOPSLA '05), October 2005, pp. 97-116.

[6] J. McC. Smith, D. Stotts, Elemental Design Patterns: A Formal Semantics for Composition of OO Software Architecture, in Proceedings of the 27th Annual IEEE/NASA Software Engineering Laboratory Workshop, Greenbelt, MD, 2002, pp. 183-190.

[7] S. Maggioni, Design Pattern Clues for Creational Design Patterns, Proceedings of the 1st International Workshop on Design Pattern Detection for Reverse Engineering (DPD4RE 2006), Benevento, Italy, October 2006.

[8] F. Arcelli, M. Zanoni, Refining Design Pattern Detection through Design Pattern Clues, submitted to the Journal of Object Systems, August 2010.