

UsCaAb : A Tool for Abstracting Use Case Diagrams

Mario Luca Bernardi, Giuseppe Antonio Di Lucca
RCOST Research Centre on Software Technology
Dept. of Engineering
University of Sannio
Palazzo ex Poste, via Traiano, 82100 Benevento, Italy
dilucca/mlbernar@unisannio.it

1. Introduction

The Use Case Abtractor (UsCaAb) tool automatically recovers use case diagrams from C++ and Java software systems. The target system is represented by the E-M-M Graph [1, 2]; hence this graph is analysed to identify groups of related components implementing the use-cases and the relationships among them. The aim of the tool is to provide software engineers with a useful and extensible environment supporting program comprehension, software maintenance and testing tasks.

2. UsCaAb architecture

The figure 1 shows the overall architecture of the UsCaAb tool. The Parsing module extracts, by static analysis of the source code, information about the control and data flow. The module includes an extractor for the Java language, based on the BCEL library [3], and an extractor for the C++ language based on CppML [4, 5]. More extractors for other different languages can be easily added. The extracted information is exploited by the E-M-M Graph Representation module for building the E-M-M Graph, while the E-M-M Graph Browser is devoted to visualise and edit the Graph itself. The Repository stores the information resulting by the computation of the other modules. The IR-Files stores an intermediate representation resulting by the static code analysis; the E-M-M Graph stores a representation of the E-M-M Graph according to a predefined meta-data model; the Configuration Files stores information about the target system to analyse; the Use Case Diagrams component stores the information about the abstracted use case diagrams according to a predefined XMI format. The Front End module includes the E-M-M Graph Browser and the Use Case Abtractor. The browser allows users to navigate through the E-M-M Graph; each element of the Graph is associated to the source code component it represents and users can visualise the code and some associated information just by clicking on it. The Use Case Abtractor analyses the E-M-M Graph to abstract a hierarchy of use case diagrams according to the rules defined in [1, 2]. The System Configuration module allows the user to configure the UsCaAb tool: some analysers' parameters characterising the target system to analyse (such as the external libraries used, the I/O statements and event handlers to consider as inport and output of the E-M-M Graph, etc.), may be set by the user. Moreover, the module allows to the user to assign a concept to each of the recovered use cases.

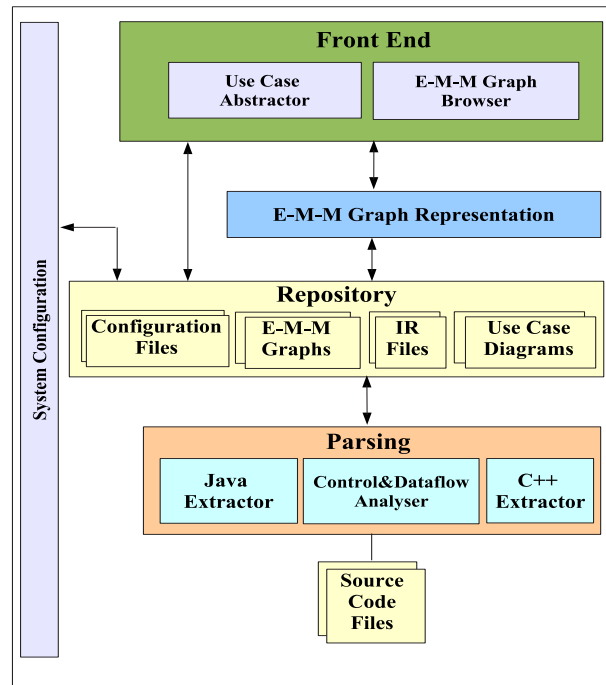


Figure 1. UsCaAb's architecture view

References

- [1] G. A. Di Lucca, A. R Fasolino, U. De Carlini 'Recovering Use Case models from Object-Oriented Code: a Thread-based Approach', Proceedings of the IEEE 7th Working Conference on Reverse Engineering, Brisbane, Queensland, Australia, Nov. 23-25, 2000, IEEE Comp. Soc. Press, Los Alamitos, California.
- [2] M. L. Bernardi, G. A. Di Lucca 'Supporting the Comprehension of Object-Oriented Software Systems by Extended M-M Graph', Proc. of IASTED Conference on SOFTWARE ENGINEERING - SE 2005, February 15-17, 2005, Innsbruck, Austria.
- [3] Apache Software Foundation, Byte Code Engineering Library (BCEL), <http://jakarta.apache.org/bcel>
- [4] Rudolf Ferenc, Arpad Beszedes, 'Towards a Standard Schema for C++', University of Szeged, Ungheria, 2001
- [5] R. Ferenc, Á. Beszédés, M. Tarkiaainen, T. Gyimóthy, 'Columbus - Reverse Engineering Tool and Schema for C++', Proc. 18th International Conference on Software Maintenance, Montréal, Canada, October 3-6, 2002, IEEE Comp. Soc. Press, Los Alamitos, California.