Analysing the hierarchical structure of Erlang applications

Bence János Szabó, Melinda Tóth, István Bozó

Eötvös Loránd University, Faculty of Informatics, Department of Programming Languages and Compilers

{szbtadi,tothmelinda,bozoistvan}@caesar.elte.hu

RefactorErl [?] is a source code comprehension and refactoring tool for Erlang [?]. The tool is a static code analyser, that is, all the information is extracted from the source code without running the program. The aim of the tool is to make the source code more understandable and easier to maintain which RefactorErl reaches by providing several analysis features and tools. The extracted information of the source code is stored in a semantic program graph (SPG) which is available for users through a query language.

The tool provides several basic semantic analysers, like function call analysis, data flow analysis, etc. There are several advanced analysers built on the basic semantic analysers, for example duplicate code analysis, dead code analysis, and dependency graphs. However, the analysis of supervisors and applications [?] of Erlang programs was not included. Both concepts are fundamental in Erlang because they describe dependencies and connections between modules.

Supervisors can monitor and manage other processes and handle errors based on described strategies. Supervisors which are built hierarchically (monitors supervisor processes) are called supervision trees.

Applications in Erlang could be viewed as components implementing some specific functionality and can be handled as a unit, therefore applications have important role in program deployment. An Erlang application consists of Erlang modules and an application descriptor file.

In this paper we are going to extend the semantic layer of the semantic program graph with information about supervisors and applications. We provide formal definitions both for the extensions of the model of the semantic program graph and for the algorithms of the semantic analyses. In addition we define two new graphs, a view of the supervision hierarchy and a view of the structure and dependence of applications.

References

[1] I. Bozó, D. Horpácsi, Z. Horváth, R. Kitlei, J. Kőszegi, M. Tejfel, and M. Tóth. Refactorerl – source code analysis and refactoring in Erlang. In *In proceeding of the 12th Symposium on Programming Languages and Software Tools, Tallin, Estonia*, 2011.

[2] Logan, M., Merritt, E., and Carlsson, R.: *Erlang and OTP in Action*, Manning Publications Co., 2010. ISBN 9781933988788.