Acta Cybernetica **14** (1999) 1-.

An Environment for Safe Refactoring Clean Programs*

Rozália Szabó-Nacsa[1], Péter Diviánszky[2], Zoltán Horváth[2]

[1] Department of Software Technology and Methodology,
Eötvös Loránd University, Hungary
e-mail: nacsa@inf.elte.hu

[2] Department of Programming Languages and Compilers,
Eötvös Loránd University, Hungary
e-mail:divip@aszt.inf.elte.hu, hz@inf.elte.hu

Abstract

We present here an interactive environment where one can incrementally carry out programmer-guided safe (meaning-preserving) program transformations in functional languages. We discuss an alternative approach to the problems of storing and extracting the syntactic and also the static semantic information in order to be flexible enough to perform the desired transformations. In our approach the program to be redesigned is stored in a relational database.

A transformation case study will help us to demonstrate how this database can be used to transform programs, check the preconditions and make compensation steps to ensure correct transformations.

We also show an interactive environment which will help the programmer to choose the appropriate refactoring step and its parameters. During redesign process the programmer is faced with one most appropriate "view" extracted from the database.

Different transformations can be carried out on different views, depending on which view is preferable for the programmer and/or which view is more suitable for the given transformation.

Categories and Subject Descriptors: D.2.3 [Software Engineering]: Coding Tools and Techniques; D.2.6 [Software Engineering]: Programming Environments; D.3.2 [Programming Languages]: Language Classifications - *Applicative (functional) languages*;

Key Words and Phrases: Clean, Haskell, program transformation, refactoring, language-aware programming environments, semantic editors

*Supported by the Hungarian National Science Research Grant (OTKA), Grant Nr. T037742 and by the Bolyai Research Scholarship.

References

[1] Li, H., Reinke, C., Thompson, S.: *Tool Support for Refactoring Functional Programs*, Haskell Workshop: Proceedings of the ACM SIGPLAN workshop on Haskell, Uppsala, Sweden, Pages: 27–38, 2003.

[2] Fóthi, Á., Horváth, Z., Nyéky-Gaizler, J.: *A Relational Model of Transformation in Programming*, Proceedings of the 3rd International Conference on Applied Informatics, Eger-Noszvaj, Hungary, Aug. 26-28, 1997. 335-349.

[3] Plasmeijer, R., Eekelen, M.: *Concurrent Clean Language Report*, Technical Report CSI-R9816, Computing Science Institute, University of Nijmegen, 1998.

[4] Fowler, M., Beck, K., Brant, J., Opdyke, W., Roberts, D.: *Refactoring: Improving the Design of Existing Code*, Addison-Wesley, 1999.

[5] *Martin Fowler's refactoring site*, www.refactoring.com

[6] de Mol, M., van Eekelen, M., Plasmeijer, R.: *SPARKLE: A Functional Theorem Prover*, International Workshop on the Implementation of Functional Languages, IFL 2001, Selected Papers, Springer-Verlag, LNCS 2312, pages 55-71.

[7] Horváth, Z., Kozsik, T., Tejfel, M.: *Verifying invariants of abstract functional objects — a case study* 6th International Conference on Applied Informatics, Eger, Hungary January 27–31, 2004.

[8] Horváth, Z., Kozsik, T., Tejfel, M.: *Extending Sparkle Core Language with Object Abstraction* The Fourth Conference of PhD Students in Computer Science, Szeged, Hungary July 1–4, 2004.

[9] Diviánszky, P., Szabó-Nacsa, R., Horváth, Z.: *Prototype Environment for Refactoring Clean Programs* 6th International Conference on Applied Informatics, Eger, Hungary January 27–31, 2004.