Extending a system with verified components[1]

Ákos Dávid, Tamás Pozsgai, and László Kozma

In [1] an outline of an educational framework was presented that is able to help students of Information Technology to build applications from a set of verified components. The testing of component-based systems can be extremely complicated because it is usually not possible for system developers to pre-check the compatibility of the individual parts before the actual integration takes place. The situation may be even worse when prefabricated components, so-called components off-the-shelf (COTS) are used as the environment of the development and the deployment are rarely the same. Even if they are the same or relatively similar, the use of these components may differ considerably from the original purpose they were developed for. Testing of a component-based solution consists of two distinct activities: testing of the components and an integration testing of the assembled application. A system cannot be considered correct if its components do not work properly. On the other hand, can the proper functioning of a system be guaranteed if its components are verified in some ways? Unfortunately, all the information on the correctness of the individual components become irrelevant and out-of-date from the moment they are used anywhere but the original environment. The solution to this problem can be based on the idea of building correct programs in which reliability is built-in. This means the correctness properties of a program may be transferable or at least checkable in other environments as well. The concept behind that is the method of "design-by-contract" introduced by Bertrand Meyer in Object-Oriented Programming (OOP) [1]. The idea of using contracts in combination with built-in testing technologies according to Hans Gerhard Gross would offer a long-waited solution to increase our confidence in third-party components [3]. Considering other alternatives model-checking seems to be applicable as an automatic technique for verifying finite state concurrent systems [4]. The main challenge in model checking is dealing with the state space explosion problem. This problem occurs in systems with many components that can make transitions in parallel. It means that extending an existing system with one or more components may cause difficulties for model checking tools by increasing the state space exponentially. In [5] open incremental model checking is introduced - and compared to traditional modular model checking methods - to address the changes to a system rather than re-checking the entire system model including the new extensions. In our paper we study the practical aspects of using open model checking by working out a sample system consisting of verified components and prove the efficiency of this new method within an educational framework.

References

[1] Á. Dávid, T. Pozsgai, and L. Kozma. Educational framework for developing applications built from verified components *Proceedings of the Ninth Symposium on Programming Languages and Software Tools*, pp. 7-18, Aug 13-14, 2005, Tartu, Estonia, published by Tartu University Press.

[2] B. Meyer. Object-Oriented Software Construction Second Edition, *Prentice Hall*, 1997.

[3] H.G. Gross. Component-Based Software Testing with UML, *Springer*, 2005.

[4] E. Clarke, O. Grumberg, and D. Peled. Model Checking, *MIT Press*, 2000.

[5] N.T. Thang and T. Katayama. Open Incremental Model Checking (Extended Abstract) *Proceedings of ACM SIGSOFT Symposium on the Foundations of Software Engineering*, Oct 31-Nov 1, 2004, Newport Beach, California, USA.

[1]This research work was supported by GVOP-3.2.2-2004-07-005/3.0.