Analysing Synchronization Contract Support in Modern Software Development Methodologies

György Orbán

The quality of the software systems must be increased. There are many techniques and methods to achieve this. In our paper we will focus one of these techniques the contract based development. Bertrand Meyer devised the expression "design by contract" which is a trademark of Eiffel Software. He created the Eiffel programming language in mind of contract support in the implementation and design. Not just a programming language, a software development methodology was created.

Four levels (syntactic, behaviour, synchronization, quality of service) of contracts can be separated based on Beugnard's categorization. At the different levels different type of contracts can be defined based on the specification. Contract based development is a continuously evolving area, but most of the developments are connected to the second, behaviour level. There are many tools which support contract based development at the implementation level. Programming languages like Eiffel, D with native support and other languages (Java, .NET, Python etc.) with extensions. There are some tools and extensions which can be integrated into an IDE and others can be used as a library. With these extensions the contracts can be weaved with different techniques into the source code or byte code. Contracts can be created at different levels in a software system. In the modern software development methodologies like Model Driven Development (MDD) the contract creation support should be at the modelling level. To achieve this we have different tools and methods. One of the most widely used modelling language is the Unified Modeling Language (UML). Object Constraint Language (OCL) expressions in the UML models can be used as contracts. With tools like DresdenOCL it is possible to generate Java source code from the OCL expressions. With this source code generation second level (behaviour) contracts can be created. But this is only one level from the four. There are many Interface Description Languages (IDL) for different types of systems and in most cases the source code generation is possible from these descriptions. The first two contract levels have more modelling and tool support.

In the paper we would like to focus on the modelling and implementation support for the synchronization (third level) contracts. From the modelling view the OCL, as a synchronization level contract definition language and from the implementation view different concurrency techniques will be examined. As a summary of our work we would like to recommend a software development process workflow with tools and methods used in the different steps to create more reliable, better quality software systems.

Acknowledgements

This research was supported by the European Union and co-financed by the European Social Fund (grant agreement no. TAMOP 4.2.1./B-09/1/KMR-2010-0003).

References

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

[2] Beugnard, A. et al., Making Components Contract Aware. *Computer*, 32(7), pp.38-45. 1999

[3] Jézéquel, J.-M., Object-oriented software engineering with Eiffel, Redwood City, CA, *Addison Wesley Longman Publishing Co., Inc.* 1996

[4] Morandi, B., Bauer, S.S. & Meyer, B., SCOOP - A Contract-Based Concurrent Object-Oriented Programming Model. *In P. Müller, ed. Advanced Lectures on Software Engineering. Berlin, Heidelberg: Springer Berlin Heidelberg,* pp. 41-90. 2010

[5] Sendall S., Strohmeier A., Specifying Concurrent System Behavior and Timing Constraints Using OCL and UML. *In Proceedings of the 4th International Conference on The Unified Modeling Language, Modeling Languages, Concepts, and Tools,* Springer-Verlag, London, UK, 391-405. 2001