Chair of Programming Methodology

Master Thesis

Verification of Design Patterns

Simon Hofer

21-10-2008

Supervisors: Prof. Peter Müller, Joseph N. Ruskiewicz

Introduction The Spec# programming system[7, 3] consists of a programming language, a compiler, and a static verifier called Boogie[1]. The programming language is an extended version of C# with method contracts, object invariants, non-null types, and an ownership model. All these constructs are used to verify programs that are sound and modular[2, 5]. Spec# is good for the verification of strictly hierarchical object collaborations, but shows specification and verification problems in non-hierarchical contexts. Since programmers use design patterns[4] for implementing object collaborations, this thesis will propose and implement methodologies and language constructs for the verification of design patterns in Spec#.

Goal of the thesis The goal of the thesis is to propose and implement verification methodologies and new Spec# language constructs for the Singleton, Visitor and Composite design patterns in order to simplify the specification process and enhance the specification expressiveness for the verification of object collaborations.

Main parts and Timeline The thesis consists of two main parts, the formal specification of verification methodologies and the implementation in Spec#.

2 weeks	Learn Spec# and background reading
3 weeks	Problem definition
6 weeks	Design methodologies
9 weeks	Implementation
6 weeks	Write the report

Possible extensions to the thesis include:

1. an extensible system for design pattern verification.
2. more design patterns.
3. the use of dynamic frames in design pattern verification.

References

[1] Mike Barnett, Bor-Yuh Evan Chang, Robert DeLine, Bart Jacobs, and K. Rustan M. Leino. Boogie: A modular reusable verifier for object-oriented programs. In *LNCS*, volume 4111. Springer, 2006.

[2] Mike Barnett, Robert DeLine, Manuel Fähndrich, K. Rustan M. Leino, and Wolfram Schulte. Verification of object-oriented programs with invariants. *Journal of Object Technology*, 3:2004, 2004.

[3] Mike Barnett, K. Rustan M. Leino, and Wolfram Schulte. The Spec# programming system: An overview. In *LNCS*, volume 3362. Springer, 2004.

[4] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.

[5] K. Rustan M. Leino and Peter Müller. Object invariants in dynamic contexts. In *ECOOP*, volume 3086. Springer, 2004.

[6] K. Rustan M. Leino and Peter Müller. Modular verification of static class invariants. In *LNCS*, volume 3582. Springer, 2005.

[7] Microsoft Research. Spec# programming system. http://research.microsoft.com/specsharp/.