Contract-Based Design: a Temporal Logics Approach*

Alessandro Cimatti
FBK-irst, Trento, Italy
cimatti@fbk.eu

Stefano Tonetta
FBK-irst, Trento, Italy
tonettas@fbk.eu

Categories and Subject Descriptors

D.2.4 [**Software Engineering**]: Software/Program Verification—*Formal methods, Programming by contract*; D.2.11 [**Software Engineering**]: Software Architectures—*Languages*

General Terms

Design, Languages, Verification

Keywords

Constract-Based Design, Temporal Logic

1. EXTENDED ABSTRACT

Contract-based design, first conceived for software specification [7] and now also applied to embedded systems (cfr. e.g., [2, 1]), structures the component properties into contracts. A contract specifies the properties assumed to be satisfied by the component environment (assumptions), and the properties guaranteed by the component in response (guarantees). There are several points supporting the idea of contract-based reasoning. The first one is that it provides a clean framework for compositional verification of global properties of a system: the contracts are used as landmarks for the proof, so that in the end it is possible to obtain the guarantee for the global property out of the proof that each of the components satisfies its contracts, and that the individual contracts entail the global property. The second is that it supports stepwise refinement, so that when a component is decomposed, the corresponding specification is decomposed at the same time, i.e. way before the behavioral descriptions are provided. The third reason is the support of component reuse: the proof of refinement holds for any component implementation satisfying the contracts of the component leaves.

In the contract framework originally proposed in [6], assumptions and guarantees are specified as temporal formulas. Checking the correctness of contracts refinement is supported by generating a set of sufficient and necessary conditions. These proof obligations are temporal logic formulas obtained from assumptions and guarantees, so that they are valid if and only if the contracts refinement is correct. The approach is applied to various temporal logic, ranging from LTL [8] discrete state, to HRELTL [5], a variant of LTL where formulas represent sets of hybrid traces, mixing discrete- and continuous-time steps, and therefore amenable to model properties of timed and hybrid systems.

The approach is implemented in OCRA [3], available at https://es.fbk.eu/tools/ocra. The contracts are specified in Othello [4], a human-readable language that can be mapped to temporal formulas. In case of LTL or the propositional fragment of HRELTL, the proof obligations can be proved valid using BDD- or SAT-based model checking techniques. In the general case of HRELTL, reasoning relies on Satisfiability Modulo Theory (SMT). Since logical entailment for HRELTL is undecidable, bounded model checking techniques are used to find counterexamples to the contract refinement [5, 6].

The approach has been developed within the European project SafeCer (http://www.safecer.eu), focusing on the compositional certification of embedded systems. The tool has been integrated within CASE tools such as CHESS, for SysML/UML-based design, and AUTOFOCUS3 (http://af3.fortiss.org/). It is currently in use by the industrial partners of the SafeCer, and has been validated within the aerospace model-based methodology of FoReVer (https://es.fbk.eu/projects/forever).

*This work is partly sponsored by the EIT project N. 13060-A1313 CPSE Tool Integration Framework Open-Source Booster.

2. REFERENCES

[1] S. Bauer, A. David, R. Hennicker, K. Larsen, A. Legay, U. Nyman, and A. Wasowski. Moving from Specifications to Contracts in Component-Based Design. In *FASE*, pages 43–58, 2012.

[2] A. Benveniste, B. Caillaud, A. Ferrari, L. Mangeruca, R. Passerone, and C. Sofronis. Multiple Viewpoint Contract-Based Specification and Design. In *FMCO*, pages 200–225, 2007.

[3] A. Cimatti, M. Dorigatti, and S. Tonetta. OCRA: A Tool for Checking the Refinement of Temporal Contracts. In *ASE*, 2013. In press.

[4] A. Cimatti, M. Roveri, A. Susi, and S. Tonetta. Validation of Requirements for Hybrid Systems: a Formal Approach. *ACM Trans. Softw. Eng. Methodol.*, 21(4):22, 2012.

[5] A. Cimatti, M. Roveri, and S. Tonetta. Requirements Validation for Hybrid Systems. In *CAV*, pages 188–203, 2009.

[6] A. Cimatti and S. Tonetta. A Property-Based Proof System for Contract-Based Design. In *SEAA*, 2012.

[7] B. Meyer. Applying "Design by Contract". *Computer*, 25(10):40–51, 1992.

[8] A. Pnueli. The temporal logic of programs. In *FOCS*, pages 46–57, 1977.