Scenario-Based Analysis of Statechart Construction

Benjamin De Leeuw[1] and Albert Hoogewijs[2]

Universiteit Gent, Galglaan 2, B-9000 Gent, België,
[1]benjamin.deleeuw@ugent.be**,
[2]albert.hoogewijs@ugent.be

1 Abstract

Scenario based specification [1] is an approach to behavioural specification of systems as inter object messaging patterns. Grafted on this theory, some ongoing work studies the construction of equivalent statecharts [2] by analyzing and composing these scenarios with *sequential*, *disjunctive*, *conjunctive* and *iterative* composition rules[3, 4]. We present a technique for statechart construction, inspired by these scenario composition rules, which moreover led to a valuable classification of statechart construction patterns. In this approach we construct statecharts by iteratively composing atoms to more complex constructs, using four statechart rewrite rules: *burst*, *bubble*, *concurrency* and *transreduction*. An atom or atomic statechart is a small one state statechart with one incoming start edge and one outgoing edge to termination, where the start edge label is a list of actions, and the label of the terminal edge consists of a trigger and a guard. Combining two of these atomic statecharts sequentially results in a statechart with two states, connected with a normal transition with trigger, guard and actionlist, and a start and terminal edge as before. The composition rules we present, abstract this sequential composition of atoms, by taking an edge or state of a statechart and replacing it with a more complex construct consisting of one or more added states and edges. The *burst* rule rewrites edges, where the *bubble*, *concurrency* and *transreduction* rules rewrite states and redistribute incoming and outgoing edges of the original state to newly generated states. The edge labels are generated accordingly and introduce demonic (event based) and angelic (shared memory based) choice in the generated statechart. We provided a Java implementation of this generation process based on four rewrite rules. Applicability of generated statechart test cases can be in any form of persuasive argument towards stakeholders in industry and the science community. With the proposed tool, testing with (randomly) generated statecharts becomes available and software enigneers will be able to more acurately evaluate the usefulness of emerging statechart analysis tools in a more "hostile" environment than can be offered by made-up ATM or vending machine specifications.

** Funded by Universiteit Gent (BOF/GOA project B/03367/01 IV1) and the Prof. Dr. Wuytack Fund.

References

1. Carroll, J.M., ed.: Scenario-Based Design: Envisioning Work and Technology in System Development. John Wiley and Sons (1995)
2. Harel, D.: Statecharts: A Visual Formalism for Complex Systems. Science of Computer Programming **8**(3) (1987) 231–274
3. Vasilache, S., Tanaka, J.: Synthesizing Statecharts from Multiple Interrelated Scenarios (2001)
4. Whittle, J., Schumann, J.: Generating Statechart Designs from Scenarios. In: ICSE '00: Proceedings of the 22nd international conference on Software engineering, New York, USA, ACM Press (2000) 314–323