Dynamic analysis of UML Statemachines

Gábor Huszerl

My main research topic is the dynamic analysis of Unified Modeling Language (UML) models for evaluation of performance, performability and timing properties. Instead of writing new analysis tools for the examination of the UML models, I am exploring the possibility of utilizing the proven ones. For performance evaluation and analysis Petri nets offer a mathematically well-defined methodology with precise semantics and a theoretical background. Petri nets are used in specification, design, verification and analysis of concurrent distributed systems for over 25 years, and there are already many Petri net tools known. I have defined a transformation of UML statemachines to Stochastic Reward Nets (SRNs), a special class of Petri nets, in special consideration of event processing, state hierarchy and priority of transitions. The transformation is defined by a set of design patterns. In order to support timing analysis I have extended the original UML models by including timing information [1, 2], and defined Petri net (SRN) patterns describing common semantics of timed and guarded state transitions. For analysing the arising SRNs well-known Petri net tools are used.

When transforming between different formalisms it is necessary to study the equivalence of the semantics of the source and target models. This paper examines a single design pattern of the above mentioned ones. This pattern implements the step semantics of the UML statemachines using SRNs.

UML [3] statemachines are hierarchical, where states can contain substates or concurrent submachines. Transitions may have their source and target states at different levels of the hierarchy. The transitions are triggered by events, and guarded by logical expressions. When several transitions are enabled, the maximal non-conflicting set of them may fire at the same time in a single step. Each step consists of the following hypothetical phases: dispatching an event, collecting the triggered and enabled transitions, selecting an appropriate subset of them and finally firing the selected transitions simultaneously.

SRNs have no hierarchy of places, and their transitions fire independently one at a time, thus the priority of transitions needs extra constructions in the patterns. These constructions are tree structures for every event type, determining which set of SRN transitions –representing individual UML transitions– may fire in the given step.

The SRN resulted by the transformation implements the phases of an above mentioned step. During the steps, there are intermediate states of the SRN, which have no valid counterpart in the statechart, however these states are transient ones vanishing before completion of the step.

This paper discusses the design patterns mapping the UML transition semantics to SRNs, and it focuses on the equivalence of the UML semantics of transition steps and the defined SRN structure. The proven equivalence of the UML model and the model directly analysed provides the designer precise information about performance, performability and timing properties of his current design.

References

[1] M. Dal Cin, Huszerl G., K. Kosmidis: *Transformation of Guarded Statecharts for Quantitative Evaluation of Dependable Embedded systems* - EWDC-10, ISBN 3-85403-125-4, pp. 143-187, Vienna, Austria, May 6-7, 1999

[2] M. Dal Cin, Huszerl G., K. Kosmidis: *Quantitative Evaluation of Dependability Critical Systems Based on Guarded Statechart Models* – In Proc. HASE'99, Washington DC, USA, 1999. November, pp. 37-45.

[3] *"UML Semantics", version 1.1* – Object Management Group, September 1997