Semantics of refinement in SCXML

Karla Morris[1], Colin Snook[2], and Thai Son Hoang[2]

[1] Sandia National Laboratories, Livermore, California, U.S.A.
[2] ECS, University of Southampton, Southampton, United Kingdom

Abstract. Formal semantics of State-Chart XML (SCXML) with refinement...

Keywords: run-to-completion, state-charts, refinement

1 Introduction

We formalise the semantics of refinement in SCXML by modelling, in Event-B,

- the structure and behaviour of abstract SCXML models,
- the structure, behaviour of refined SCXML models and their relationship to abstract models.
- We then refine the latter to remove verification artifacts and show that it is no different to the abstract model and hence SCXML refinement can be applied iteratively.

Since SCXML consists of a state-chart behaviour superimposed with a 'run to completion' execution semantics, we formalise the refinement of these two parts separately and then compose the two definitions to obtain the complete SCXMl refinement semantics.

2 Refinement in SCXML

What can we do in a refinement

- add nested state-machine
- strengthen guard of transition to new sub-state
- strengthen action of transition to new sub-state
- add new external trigger
- add new internal trigger (can be raised non-determinstically or by a transition
- add new triggered or untriggered transition
-

what we can not do

- cannot change the triggering of a refined transition - i.e. if it is untriggered it always will remain untriggered and if triggered, that trigger will never be changed.
- refined transitions cannot raise more internal triggers except newly introduced ones. I.e. when the trigger is first introduced, transitions that do not raise it, never will. (has to do with finalisation)

3 The semantics of refining State-machines in general

Colin: Describe the State-machine semantics model - note that we simplified.. we did not deal with parallel regions

4 The semantics of refining an SCXML Run to Completion

To define the SCXML run to completion execution we first define the static elements involved: Triggers are partitioned into either Internal or External triggers partition (TRIGGERS, InternalTriggers, ExternalTriggers, {nullTrigger}) A null trigger nullTrigger is introduced for convenience to simplify the definition of the initial condition: the execution begins by firing untriggered transitions which is otherwise only done after a trigger is consumed. Hence the null Trigger is said to be consumed during initialisation.

We define the internal and external trigger queues as subsets of internal, resp. external triggers $\mathsf{InternalQueue} = \mathbb{P}(\mathsf{InternalTriggers})$ $\mathsf{ExternalQueue} = \mathbb{P}(\mathsf{ExternalTriggers})$

Colin: THIS NEEDS COMPLETING

5 The semantics of refining SCXML models

Colin: This is some old text.. needs updating for our latest approach

To model the complete State-Chart XML (SCXML) refinement semantics we now combine the previous two models. That is, using the inclusion mechanism built into CamilleX, we combine our models of scxml execution with our models for state-machines.

5.1 M1 - refinement of combined scxml and state-machine

For this stage we include the corresponding M1 stages for SCXML and state-machine.

The gluing context defines the needed refinement relationship between concrete and abstract parts of the model.

1. For transitions that refine abstract transitions, the transition is triggered if and only if its corresponding abstract transition is triggered.
2. For *triggered* transitions that refine abstract transitions, the trigger is the same as that of the corresponding abstract transition.
3. For *finalised* transitions that refine abstract *finalised* transitions, the transition source is the same as that of the corresponding abstract transition. I.e. you cannot strengthen the source of an already finalised transition.
4. the abstract *finalised* transitions are a subset of the abstract transitions that correspond to concrete finalised transitions. I.e. once transitions are finalised they stay finalised in refinements.

Initially, for item 1 we defined a single axiom giving the equality of the domains of the abstract and concrete transition-trigger relationship. However, this was insufficient to prove the guard strengthening of the untriggered refined transitions because the transition_link relationship is not injective. (I.e. in general, an abstract transition could be refined by more than one concrete transition) Instead, we defined two axioms, universally quantifying over the set of refined transitions that are/are not triggered, with the consequent that the corresponding abstract transitions are/are not triggered. These axioms were sufficient to automatically discharge the guard strengthening POs for both triggered and untriggered refined transitions.

5.2 M2 - merging old and new events

We found a problem when we tried to merge the refined and new transitions of the combined scxml and statechart semantics. The problem occurs for merging both triggered and untriggered transitions. The problem is that we have not considered the case of merging a new state-machine transition with an old refined scxml triggered transition. Probably this is nonsensical and we should add guards to prevent it. Ok for triggered but for untriggered our way of distinguishing refined scxml untriggered transitions is via the disappearing variable dqaux

6 Conclusion

In conclusion ...

All data supporting this study are openly available from the University of Southampton repository at https://doi.org/....tbd

Acknowledgements Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.