Refinement and Verification of Responsive Control Systems

Authors:

K. Morris[1], C. Snook[2], T.S. Hoang[2],

G. Hulette[1], R. Armstrong[1], and M. Butler[2]

[1] Sandia National Laboratories

[2] Southampton University

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.

OUTLINE

- Introduction
  - Motivation
  - Background
    - SCXML
    - Event-B
    - UML-B Statemachine
  - Run-to-Completion
- Statechart Refinement
- SCXML Model Translation to Event-B
- Verification of Control Responses
- Conclusions

Sandia National Laboratories

UNIVERSITY OF Southampton

School of Electronics and Computer Science

Introduction

Motivation

- Graphical languages are attractive to engineers (e.g. statecharts generalization of statemachines)

- **Rodin/Event-B** support abstraction and refinement and enable the use of formal proofs in earlier stages of the design

- There are many definitions of **refinement**, we are concerned with a refinement that preserves **safety properties** from one refinement level to the next

- To enable our form of refinement statechart transitions cannot cross state containment boundaries arbitrarily.

State Chart XML (SCXML)

- Modeling language based on Harel statechart
- Follow a run-to-completion semantics
  - Trigger events may enable transitions
  - Trigger events are queued whenever a trigger is raised
  - A trigger event is de-queued and consumed by firing all transitions that it enables
  - Enabled un-triggered transitions are taken repeatedly until none are enabled, and then the next trigger is de-queued
- Internal triggers are raised by transitions within the model
- External triggers are raise by the environment (non-deterministically)
- Internal triggers have higher priority than external triggers

```
<?xml version="1.0" encoding="UTF-8"?>
<scxml xmlns="http://www.w3.org/2005/07/scxml"
  xmlns:iumlb="urn:xmlns:ac.soton.uk:iumlb"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  initial="OFF"
  name="drone" version="1.0" iumlb:refinement="0">
  <datamodel>
    <data expr="::0..100" id="charge" iumlb:type="0..100" iumlb:refinement="2"/>
    ...
  </datamodel>
  ...
<state id="BATTERYOK">
  <transition event="decreaseCharge" target="BATTERYOK" iumlb:label="BATTERYOK_BATTERYOK">
    <iumlb:guard predicate="charge>=10" name="update_charge"/>
    <assign expr="charge - 10" location="charge" iumlb:refinement="2"/>
  </transition>
  <transition target="BATTERYLOW" iumlb:label="BATTERYOK_BATTERYLOW" iumlb:finalised="2">
    <iumlb:guard predicate="charge&lt;=20" name="check_charge"/>
    <raise event="toLand"/>
  </transition>
  <iumlb:invariant predicate="charge>20" name="check_charge"/>
</state>
<state id="BATTERYLOW"/>
...
</scxml>
```

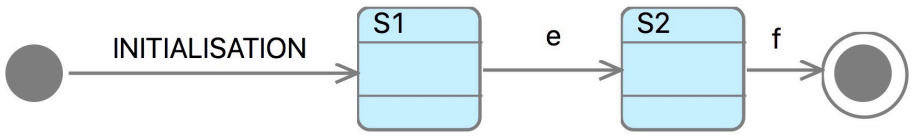
Event-B

```
machine M0
variables v
invariants I(v)
events
  INITIALISATION: begin S0(v) end
  e: any t where G(t,v) then S(t,v) end
end
```

```
machine M1
refines M0
variables v, w
invariants J(v,w)
events
  INITIALISATION: begin R0(v,w) end
  f: refines e any t where H(t,v,w) then R(t,v,w) end
end
```

- Formal method for system design via refinement
- Event-B models have two parts:
  - **Contexts:** Contain sets, constants and axioms
  - **Machines:** Variables, events, and invariants
- Machines are refined to add more details to the model
  - **Superposition refinement:** including additional variables
  - **Data refinement:** replacing abstract variables with new concrete variables
  - **New events:** Refines implicit abstract event that does nothing "skip"

UML-B




```
1 variables S1 S2
2 invariants
3  TRUE ∈ {S1, S2} ⇒ partition({TRUE}, {S1}∩{TRUE}, {S2}∩{TRUE})
4  //At most one of the states is TRUE at the time
5 events
6   INITIALISATION: begin S1, S2 := TRUE, FALSE end
7   e: when S1 = TRUE then S1, S2 := FALSE, TRUE end
8   f: when S2 = TRUE then S2 := FALSE end
9 end
```

- Diagrammatic modeling notation for Event-B
- Semantics is not UML nor run-to-completion
- Only untriggered transitions
- Encoding of State :
  - Boolean variables
  - Enumerated type
- Refinements:
  - **Superposition refinement:** add nested statemachines
  - **Guard strengthening:** new guards concerning auxiliary variable
  - **New Events:** add self transitions

Run-to-Completion

- Semantics is specified by an abstract basis that is extended by the model
- The basis consist of an Event-B context and machine
- Basis Context

```
context
  basis_c  // (generated for SCXML)
sets
  SCXML_TRIGGER  // all possible triggers
constants
  FutureInternalTrigger  // all possible internal triggers
  FutureExternalTrigger  // all possible external triggers
axioms
  partition(SCXML_TRIGGER, FutureInternalTrigger, FutureExternalTrigger)
end
```

- Basis Machine

```
VARIABLES
  iQ   //  internal trigger queue
  eQ   //  external trigger queue
  uc   //  run to completion flag
  dt   //  dequeued trigger for this run
INVARIANTS
  typeof_iQ  :  iQ ⊆ FutureInternalTrigger   //  internal trigger queue
  typeof_eQ  :  eQ ⊆ FutureExternalTrigger   //  external trigger queue
  disjointQueues  :  iQ ∩ eQ = ∅   //  queues are disjoint
  typeof_uc  :  uc ∈ BOOL   //  completion flag
  typeof_dt  :  dt ⊆ SCXML_TRIGGER   //  dequeued triggers
  oneDequeuedTrigger  :  dt≠∅ ⇒(∃t·dt={t})   //  at most one dequeued trigger
```

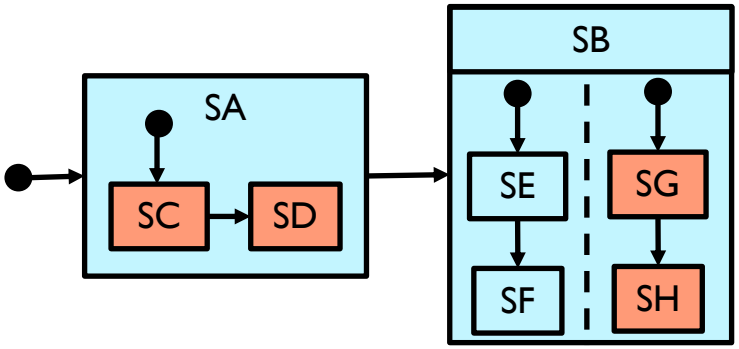
Sandia National Laboratories

UNIVERSITY OF Southampton
School of Electronics and Computer Science

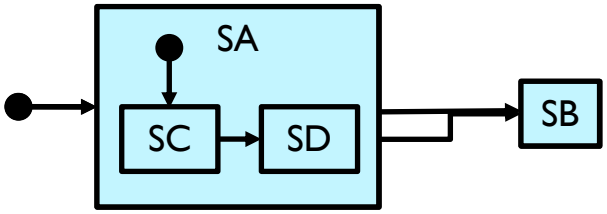
Statechart Refinement Rules

Statechsrt Refinement Rules

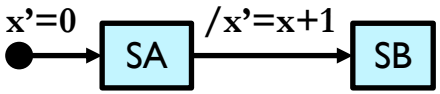
❖**Nesting Rule :** A statechart can be embedded within a state of another statechart



❖**Guard Strengthening:** Guard conditions on a transition can be strengthened



❖**Action Strengthening:** Transitions can have additional actions



Sandia National Laboratories

UNIVERSITY OF Southampton

School of Electronics and Computer Science

SCXML Model Translation to Event-B

SCXML Model Translation to Event-B

- SCXML models are first translated to UML-B and subsequently to Event-B
  - Generation of the basis machine as context that captures generic run-to-completion semantic details
  - Generation of events that correspond to all possible combination of each set of transitions that can fire together
- Model translation tool extensions
  - **Trigger queues in basis:** The encoding of trigger queues in the abstract basis machine is implemented using a queue allow used triggers to be discarded.
  - **Finalisation:** Transitions can be flagged as finalised which means their guards can not be strengthened in subsequent refinements.
  - **Restricted raising of internal triggers:** Once a trigger is introduced it must immediately be raised at that refinement level by any transitions that wish to do so.
  - **Context instantiation:** The axioms of the basis context, that allow future triggers to be added, have been improved so that ProB can automatically create an instantiation.

Sandia National Laboratories

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Verification of Control Reponses

Verification of Control Responses Using LTL Model Checking

- General form for liveness properties

$$G([\text{external\_trigger\_event}] \Rightarrow F\{\text{predicate}\}) \ ,$$

- General liveness property requires the assumption of strong fairness for all events in the model

$$SF[e1] \wedge SF[e2] \ldots \Rightarrow G([\text{external\_trigger\_event}] \Rightarrow F[\text{predicate}])$$

Sandia National Laboratories

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Conclusions and Future Work

Conclusions

- We have demonstrated a technique for introducing refinement of reactive statecharts that can be translated into Event-B for verification.

- Invariant properties about the expected coordination of states can be verified with the Rodin theorem prover. Because of the reactive nature of the system these are interpreted to hold only after the reaction is completed.

- We use the LTL model checker to verify expected reactions to environmental triggers. Another kind of liveness property that would be useful to verify is that the 'run' converges to completion.

Future Work

- Enable the verification of liveness properties using the Roding theorem prover instead of the LTL model checker
- Extend the tool support to enable automatic execution of the run to completion.
- Formalize the semantics of the extended SCXML and refinement rules

Sandia National Laboratories

UNIVERSITY OF Southampton

School of Electronics and Computer Science

QUESTIONS