Contents & Goals

Last Lecture:
• Hierarchical State Machines
• Later: Remaining pseudo-states, such as shallow/deep history; active vs. passive; behavioural features.

This Lecture:
• Educational Objectives:
  - Capabilities for following tasks/questions.
  - What does this LSC mean?
  - Are this UML models' state machines consistent with the interactions?
  - Please provide a UML model which is consistent with this LSC.
  - What is: activation, hot/cold condition, pre-chart, etc.?
• Content:
  - Reflective description of behaviour.
  - LSC concrete and abstract syntax.
  - LSC intuitive semantics.
  - Symbolic Büchi Automata (TBA) and its (accepted) language.

Course Map

Motivation: Reflective, Dynamic Descriptions of Behaviour

Recall: Constructive vs. Reflective Descriptions
[Har, 1997]

Proposes to distinguish constructive and reflective descriptions:
• "A language is constructive if it contributes to the dynamic semantics of the model. That is, its constructs contain information needed in executing the model or in translating it into executable code.
  - A constructive description tells how things are computed (which can be desired or undesired).
• "Other languages are reflective or assertive, and can be used by the system model to capture parts of the thinking that go into building the model–behavior included–, to derive and present views of the model, statically or during execution, or to set constraints on behavior in preparation for verification.
  - A reflective description tells what shall or shall not be computed.

Note: No sharp boundaries!
Mathematics

\[ N, E, f \]

\[ N \in 1 \rightarrow 2 \rightarrow \ldots \]

\[ \sigma(\ldots) \rightarrow \sigma(\ldots) \rightarrow \sigma(\ldots) \rightarrow \sigma(\ldots) \rightarrow \ldots \rightarrow \quad \]

\[ \pi \in \pi \rightarrow \pi \rightarrow \pi \rightarrow \ldots \rightarrow \quad \]

Why LSC, relation LSCs/UMLSDs, other kinds of interactions:

Recall: What is a Requirement?
Example: What is Required?

Environment:

- LightsCtrl: Operational [1, 3]
- CrossingCtrl: t(10) t
- BarrierCtrl: [1, 5] sec req lights on barrier down lights ok barrier ok ¬MvUp done

Building Blocks

- Instance Lines:
  - Environment

- Messages: (asynchronous or synchronous/instantaneous)

- Conditions and Local Invariants: (expr1, expr2, expr3 ∈ ExprS)

Intuitive Semantics: A Partial Order on Simclasses

(i) Strictly After:

(ii) Simultaneously: (simultaneous region)

(iii) Explicitly Unordered: (co-region)

Intuition:

A computation path violates an LSC if the occurrence of some events doesn't adhere to the partial order obtained as the transitive closure of (i) to (iii).
The scenario has

• the BarrierCtrl shall reply with 'barrier ok' within 1–3 time units,

• it shall finally send 'lights ok' to the lights Ctrl and BarrierCtrl, it shall be operational when receiving that event,

• the CrossingCtrl has consumed a 'secreq' event whenever the lights Ctrl has said so. They don't say they don't say it doesn't apply; maybe there's another LSC for that case. The rest of this scenario doesn't apply; maybe there's another LSC for that case.

When receiving that event, the operational CrossingCtrl activates the BarrierCtrl, which is initially in the activation mode (AM)

\[ \text{AM: invariant } I : \text{strict.} \]

\[ \text{Expr: Activation condition } (AC \in \text{expr}) \]

\[ \text{Cons: (SD = 1 \land \text{BarriersCtrl}) \rightarrow (\text{SM = 1}) \}

\[ \text{LSC: CourseMap} \]

\[ \text{ModelInstances: DS, C, T, SD, SD, SD, SD} \]

\[ \text{S, \epsilon, \sigma, \tau, \delta, \beta, \alpha, \theta, \varphi, \chi, \psi, \Omega} \]

\[ \text{down to lights Ctrl and BarrierCtrl,} \]

\[ \text{the CrossingCtrl has consumed a 'secreq' event whenever the lights Ctrl has said so. They don't say they don't say it doesn't apply; maybe there's another LSC for that case. The rest of this scenario doesn't apply; maybe there's another LSC for that case.} \]


