Contents & Goals

Last Lecture:
• Statemachinesyntax
• corestatemachines

This Lecture:
• Educational Objectives:
  Capabilities for following tasks/questions.
  • What does this StateMachine mean? What happens if I inject this event?
  • Can you please model the following behaviour.
  • What is: Signal, Event, Ether, Transformer, Step, RTC.
• Content:
  • The basic causality model
  • Ether
  • SystemConfiguration, Transformer
  • Examples for transformer
  • Run-to-completion Step

6.2.3 The Basic Causality Model
[OMG,2007b,12]
"'Causality model' is a specification of how things happen at runtime [...].
The causality model is quite straightforward:
• Objects respond to messages that are generated by objects executing communication actions.
• When these messages arrive, the receiving objects eventually respond by executing the behavior that is matched to that message.
• The dispatching method by which a particular behavior is associated with a given message depends on the higher-level formalism used and is not defined in the UML specification (i.e., it is a semantic variation point).
The causality model also subsumes behaviors invoking each other and passing information to each other through arguments of the invoked behavior, [...]. This purely 'procedural' or 'process' model can be used by itself or in conjunction with the object-oriented model of the previous example."

15.3.12 StateMachine
[OMG,2007b,563]
• Event occurrences are detected, dispatched, and then processed by the statemachine, one at a time.
• The semantics of event occurrence processing is based on the run-to-completion assumption, interpreted as run-to-completion processing.
• Run-to-completion processing means that an event [...] can only be taken from the pool and dispatched if the processing of the previous [...] is fully completed.
• The processing of a single event occurrence by a state machine is known as a run-to-completion step.
• Before commencing on a run-to-completion step, a state machine is in a stable state configuration with all entry/exit/internal-activities (but not necessarily do-activities) completed.
The same conditions apply after the run-to-completion step is completed.
• Thus, an event occurrence will never be processed [...] in some intermediate and inconsistent situation.
• [IOW,] The run-to-completion step is the passage between two state configurations of the statemachine.
The run-to-completion assumption simplifies the transition function of the StM, since concurrency conflicts are avoided during the processing of event, allowing the StM to safely complete its run-to-completion step.
• The order of dequeuing is not defined, leaving open the possibility of modeling different priority-based schemes.
• Run-to-completion may be implemented in various ways. [...]"
And?

We have to formally define what event occurrence is.

We have to define where events are stored—what the event pool is.

We have to explain how transitions are chosen—"matching".

We have to explain what the effect of actions is—on state and event pool.

We have to decide on the granularity—micro-steps, steps, run-to-completion steps (aka. super-steps)?

We have to formally define a notion of stability and RTC-step completion.

And then: hierarchical state machines.

Roadmap: Chronologically

(i) What do we (have to) cover?

UML State Machine Diagrams

Syntax.

(ii) Def.: Signature with signals.

(iii) Def.: Core state machine.

(iv) Map UML State Machine Diagrams to core state machines.

Semantics: The Basic Causality Model

(v) Def.: Ether (aka. event pool)

(vi) Def.: System configuration.

(vii) Def.: Event.

(viii) Def.: Transformer.

(ix) Def.: Transition system, computation.

(x) Transition relation induced by core state machine.

(xi) Def.: Step, run-to-completion step.

(xii) Later: Hierarchical state machines.

Ether: Examples

• A (single, global, shared, reliable) FIFO queue is an ether:

  • \( \text{Eth} \)

  • \( \text{ready} \):

  • \( \oplus \):

  • \( \ominus \):

  • \( [\cdot] \):

• One FIFO queue per active object is an ether.

• Lossy queue.

• One-place buffer.

• Priority queue.

• Multi-queues (one per sender).

• Trivial example: sink, "black hole".
For convenience require: there is $\in E$ params (i.e. $\sigma$).

$M$ states of core state machine $\sigma$ (i.e. $\sigma_0$, namely implicit attributes $\sigma_0$, values for a number of explicit attributes $\sigma_0$).

References

Let $\sigma, \epsilon$ be a pair.

A structure of a request consisting of a sender and a target.

A synchronous event may, for example, cause a state machine to trigger a transition.

The order of dequeuing is $\sigma, \epsilon$.

Run-to-completion may be implemented.

...
