Theoretical Foundations of the UML

Lecture 18: Statecharts Semantics (1)

Joost-Pieter Katoen

Lehrstuhl für Informatik 2
Software Modeling and Verification Group

`http://moves.rwth-aachen.de/teaching/ws-1415/uml/`

25. Januar 2015

RWTHAACHEN UNIVERSITY

Outline

Overview

1. Formal Definition of Statecharts

2. A Semantics for Statecharts
   - Intuition and Assumptions
   - States and Configurations
   - Enabledness
   - Consistency
   - Priority

What are Statecharts?

$$\begin{aligned} \text{Statecharts} := {} & \text{Mealy machines} \\ & + \text{State hierarchy} \\ & + \text{Broadcast communication} \\ & + \text{Orthogonality} \end{aligned}$$

Statecharts

Definition (Statecharts)

A statechart SC is a triple $(N, E, Edges)$ with:

1. N is a set of nodes (or: states) structured in a tree
2. E is a set of events
   - pseudo-event $after(d) \in E$ denotes a delay of $d \in \mathbb{R}_{\geqslant 0}$ time units
   - $\perp \notin E$ stands for "no event available"
3. $Edges$ is a set of (hyper-) edges, defined later on.

Definition (System)

A system is described by a finite collection of statecharts $(SC_1, \ldots, SC_k)$.

Tree structure

Function *children*

Nodes obey a tree structure defined by function $\mathit{children} : N \rightarrow 2^N$ where $x \in \mathit{children}(y)$ means that x is a child of y, or equivalently, y is the parent of x.

Ancestor relation $\trianglelefteq$

The partial order $\trianglelefteq \subseteq N \times N$ is defined by:

- $\forall x \in N. \, x \trianglelefteq x$
- $\forall x, y \in N. \, x \trianglelefteq y$ if $x \in \mathit{children}(y)$
- $\forall x, y, z \in N. \, x \trianglelefteq y \wedge y \trianglelefteq z \Rightarrow x \trianglelefteq z$

$x \trianglelefteq y$ means that x is a descendant of y, or equivalently, y is an ancestor of x. If $x \trianglelefteq y$ or $y \trianglelefteq x$, nodes x and y are ancestrally related.

Root node

There is a unique root with no ancestors, and $\forall x \in N. \, x \trianglelefteq \text{root}$.

Functions on nodes

The type of nodes

Nodes are typed, $type(x) \in \{ \text{BASIC}, \text{AND}, \text{OR} \}$ such that for $x \in N$:

- $type(\text{root}) = \text{OR}$
- $type(x) = \text{BASIC}$ iff $children(x) = \varnothing$, i.e., x is a leaf
- $type(x) = \text{AND}$ implies $(\forall y \in children(x).\ type(y) = \text{OR})$

Default nodes

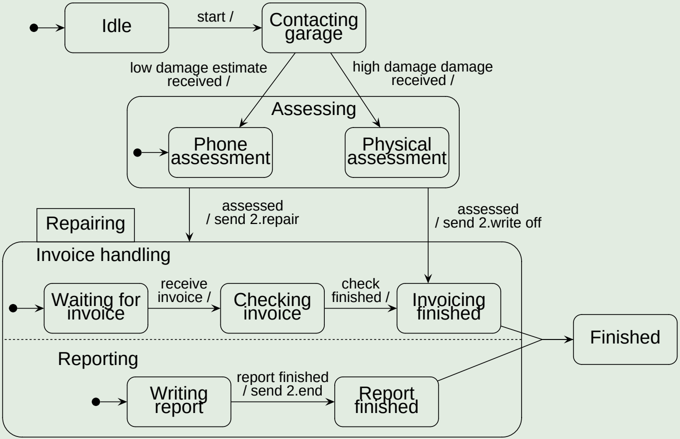
$default : N \rightarrow N$ is a partial function on $\{ x \in N \mid type(x) = \text{OR} \}$ with

$$default(x) = y \quad \text{implies} \quad y \in children(x).$$

The function *default* assigns to each OR-node x one of its children as default node that becomes active once node x becomes active.

Example

A damage assessor



Edges

Definition (Edges)

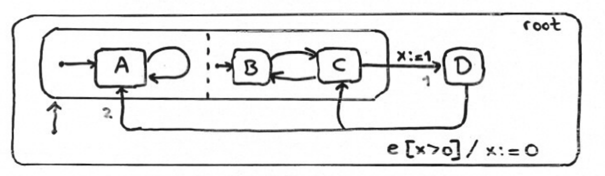
An edge is a quintuple (X, e, g, A, Y), denoted $X \xrightarrow{e[g]/A} Y$ with:

- $X \subseteq N$ is a set of source nodes with $X \neq \varnothing$
- $e \in E \cup \{ \perp \}$ is the trigger event
- $A \subseteq \mathit{Act}$ is a finite set of actions
  - such as $v :=$ expr for local variable v and expression expr
  - or *send* $j.e$, i.e., send event e to statechart SC_j
- Guard g is a Boolean expression over all variables in $(SC_1, \ldots, SC_k)$
- $Y \subseteq N$ is a set of target nodes with $Y \neq \varnothing$

The sets X and Y may contain nodes at different depth in the node tree.

Example

Example statechart



$$\text{edge 1: } \{ C \} \xrightarrow{\perp[true]/\{ x:=1 \}} \{ D \}$$

$$\text{edge 2: } \{ D \} \xrightarrow{e[x>0]/\{ x:=0 \}} \{ A, C \}$$

Overview

1. Formal Definition of Statecharts

2. A Semantics for Statecharts
   - Intuition and Assumptions
   - States and Configurations
   - Enabledness
   - Consistency
   - Priority

Towards a Statechart semantics

- Formal semantics: map $(SC_1, \ldots, SC_k)$ onto a single Mealy machine
- This is done using a step semantics distinguishing macro and micro steps
- Macro steps are "observable" and are subdivided into a finite number of micro steps that cannot be prolonged
- In a macro step, a maximal set of edges is performed
- Events generated in macro step n are only available in macro step $n{+}1$
  - If such event is not "consumed" in step $n{+}1$, it dies, and is not available in step $n{+}2$, $n{+}3$, …

Assumptions [Eshuis & Wieringa, 2000]

- Input to a macro step is a set of events (and not a queue)

  the order of event generation is ignored, i.e., if e and e' are generated in macro step i, the order in which they are generated is irrelevant in step $i+1$

- A macro step reacts to all available events

  events can only be used in macro step immediately following their generation

- Instantaneous edges and actions

- Unlimited concurrency

  there is no limit on the number of events that can be consumed in a macro step

- Perfect communication, i.e., messages are not lost

What does a single StateChart mean?

Intuitive semantics as a transition system:

- State = a set of nodes ("current control") + the values of variables
- Edge is enabled if guard holds in current state
- Executing edge $X \xrightarrow{e[g]/A} Y$ = perform actions A, consume event e
  - leave source nodes X and switch to target nodes Y

  $\Rightarrow$ events are unordered, and considered as a set
- Principle: execute as many edges at once (without conflict)

  $\Rightarrow$ the total execution of such maximal set is a macro step

States and configurations

Definition (Configuration)

A configuration of $SC = (N, E, Edges)$ is a set $C \subseteq N$ of nodes satisfying:

- root $\in C$
- $x \in C$ and $type(x) = \text{OR}$ implies $|children(x) \cap C| = 1$
- $x \in C$ and $type(x) = \text{AND}$ implies $children(x) \subseteq C$

Let $Conf$ denote the set of configurations of SC.

Definition (State)

State of $SC = (N, E, Edges)$ is a triple (C, I, V) where

- C is a configuration of SC
- $I \subseteq V$ is the set of events to be processed
- V is a valuation of the variables.

Example

Enabling of an edge

Definition (Enabledness)

Edge $X \xrightarrow{e[g]/A} Y$ is enabled in state (C, I, V) whenever:

- $X \subseteq C$, i.e. all source nodes are in configuration C
- $(\underbrace{(C_1, \ldots, C_n)}_{\text{configurations}}, \underbrace{(V_1, \ldots, V_n)}_{\text{variable valuations}}) \models g$, i.e., guard g is satisfied
- either $e \neq \perp$ implies $e \in I$, or $e = \perp$

Let $En(C, I, V)$ denote the set of enabled edges in state (C, I, V).

Macro steps

- On receiving an input e, several edges in SC may become enabled
- Then, a maximal and consistent set of enabled edges is taken
- If there are several such sets, choose one nondeterministically
- Edges in concurrent components can be taken simultaneously
- But edges in other components cannot; they are inconsistent
- To resolve nondeterminism (partly), priorities are used

Consistency: examples

To define consistency formally, we need some auxiliary concepts

Least common ancestor

Definition (Least common ancestor)

For $X \subseteq N$, the least common ancestor, denoted $lca(X)$, is the node $y \in N$ such that:

$$(\forall x \in X.\, x \trianglelefteq y) \quad \text{and} \quad \forall z \in N.\, (\forall x \in X.\, x \trianglelefteq z) \text{ implies } y \trianglelefteq z.$$

Intuition

Node y is an ancestor of any node in X (first clause), and is a descendant of any node which is an ancestor of any node in X (second clause).

Orthogonality of nodes

Definition (Orthogonality of nodes)

Nodes $x, y \in N$ are orthogonal, denoted $x \bot y$, if

$$\neg(x \trianglelefteq y) \quad \text{and} \quad \neg(y \trianglelefteq x) \quad \text{and} \quad \mathit{type}(\mathit{lca}(\{\, x, y \,\})) = \textsc{and}.$$

Orthogonality captures the notion of independence. Orthogonal nodes can execute enabled edges independently, and thus concurrently.

Scope

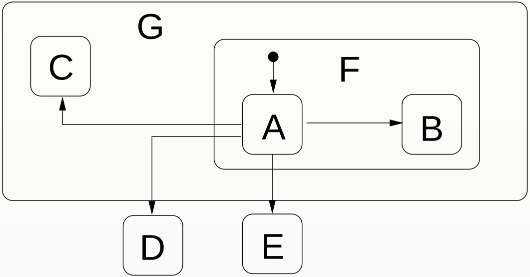
Definition (Scope of edge)

The scope of edge $X \overset{\cdots}{\longrightarrow} Y$ is the most nested $\textsc{or}$-node that is an ancestor of both X and Y.

Intuition

The scope of edge $X \overset{\cdots}{\longrightarrow} Y$ is the most nested $\textsc{or}$-node that is unaffected by executing the edge $X \overset{\cdots}{\longrightarrow} Y$.

Scope: example



$$scope(A \to D) = \text{root} \quad \text{and} \quad scope(A \to C) = G \quad \text{and} \quad scope(A \to B) = F$$

Consistency: formal definition

Definition (Consistency)

1. Edges $ed, ed' \in Edges$ are consistent if:

$$ed = ed' \quad \text{or} \quad scope(ed) \perp scope(ed').$$

2. $T \subseteq Edges$ is consistent if all edges in T are pairwise consistent. $Cons(T)$ is the set of edges that are consistent with all edges in $T \subseteq Edges$

$$Cons(T) = \{ed \in Edges \mid \forall ed' \in T : ed \text{ is consistent with } ed'\}$$

Example

On the black board.

What is now a macro step?

A macro step is a set T of edges such that:

- all edges in step T are enabled
- all edges in T are pairwise consistent, that is:
  - they are identical or
  - scopes are (descendants of) different children of the same AND-node
- enabled edge ed is not in step T implies

  there exists $ed' \in T$ such that ed is inconsistent with ed', and
  the priority of ed' is not smaller than ed
- step T is maximal (wrt. set inclusion)

Priorities

Priorities restrict (but do not abandon) nondeterminism between multiple enabled edges.

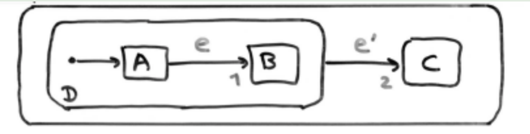
Definition (Priority relation)

The priority relation $\preceq \subseteq Edges \times Edges$ is a partial order defined for $ed, ed' \in Edges$ by:

$$ed \preceq ed' \quad \text{if} \quad scope(ed') \trianglelefteq scope(ed)$$

So, ed' has priority over ed if its scope is a descendant of ed's scope.

Example:



$2 \preceq 1$ since $scope(1) = D \trianglelefteq scope(2) = \text{root}$.

Priority: examples

Nondeterminism

Priorities rule out some nondeterminism, but not necessarily all.

What is now a macro step?

A macro step is a set T of edges such that:

- all edges in step T are enabled
- all edges in T are pairwise consistent
  - they are identical or
  - scopes are (descendants of) different children of the same AND-node
- step T is maximal (wrt. set inclusion)
  - T cannot be extended with any enabled, consistent edge
- priorities: enabled edge ed is not in step T implies

  $\exists ed' \in T.\ (ed \text{ is inconsistent with } ed' \wedge \neg(ed' \preceq ed))$

A macro step — formally

A macro step is a set T of edges such that:

- enabledness: $T \subseteq En(C, I, V)$
- consistency: $T \subseteq Cons(T)$
- maximality: $En(C, I, V) \cap Cons(T) \subseteq T$
- priority: $\forall ed \in En(C, I, V) - T$ we have
  $(\exists ed' \in T.\ (ed \text{ is inconsistent with } ed' \wedge \neg(ed' \preceq ed)))$

Note:

The first three points yield: $T = En(C, I, V) \cap Cons(T)$.

Computing the set T of macro steps in state (C, I, V)

function $nextStep(C, I, V)$

$T := \varnothing$

while $T \subset \; En(C, I, V) \cap Cons(T)$

do let $ed \in High\,((En(C, I, V) \cap Cons(T)) - T)\,;$

$\quad T := T \cup \{ed\}$

od

return T.

where $High(T) \; = \; \{ed \in T \mid \neg(\exists ed' \in T.\, ed \preceq ed')\}$