

2IW05 – Software Specification

Statecharts - Part II: Semantics

Mohammad Mousavi

Formal Systems Analysis (FSA)
Model-Driven Software Engineering
Department of Computer Science
TU/Eindhoven

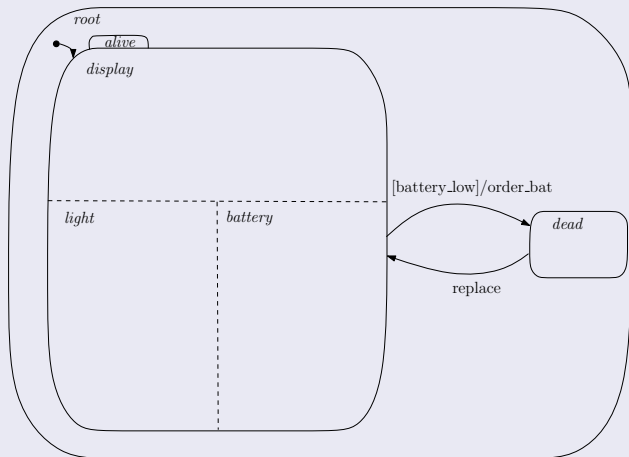
December 13, 2011

Outline

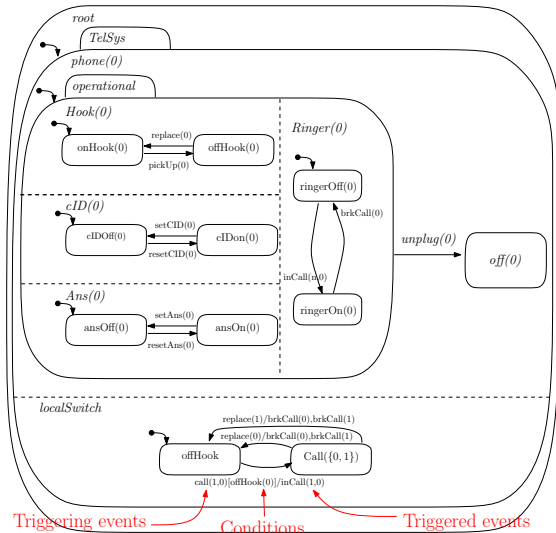
- 1 Recap
- 2 Micro-Steps
- 3 Macro-Steps
- 4 Other Features
- 5 UML State Diagrams
- 6 Conclusions

Watch

State Hierarchy



Telephone System



Transitions

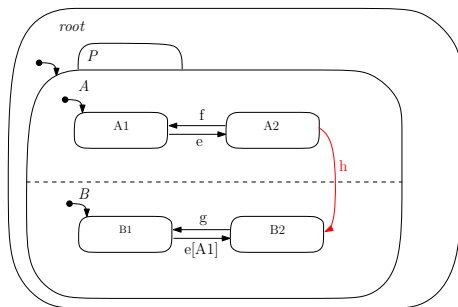
- E (events): set of events.
- C (conditions): subsets of states.
 $C = \mathbb{P} S$.
- L (labels): triples of subset of events (triggers), condition, and subset of events (generated events), usually denoted by $a[c]/e$.
 $L = (\mathbb{P} E) \times C \times (\mathbb{P} E)$.
- Tr (transitions): triples of non-empty set of source states, label, non-empty set of target states.
 $Tr \subseteq (\mathbb{P} S) \times L \times (\mathbb{P} S); \forall (s, l, s') : Tr \bullet s \neq \emptyset \wedge s' \neq \emptyset$.

Configuration

Given a state $s \in S$, a set of states $C \subseteq sdesc(s) \cup \{s\}$ is a **configuration w.r.t. s** when it is the minimal set satisfying:

- $s \in C$;
- $\forall s' \in C \text{ type}(s') = \textit{Or} \Rightarrow \exists_1 s'' : \textit{child}(s') \bullet s'' \in C$;
- $\forall s' \in C \text{ type}(s') = \textit{And} \Rightarrow \textit{child}(s') \subseteq C$.

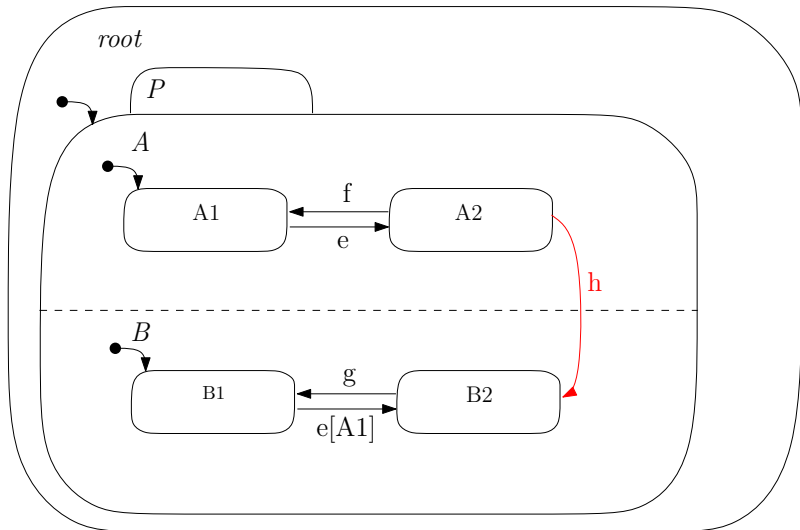
Semantical Puzzel: Configuration



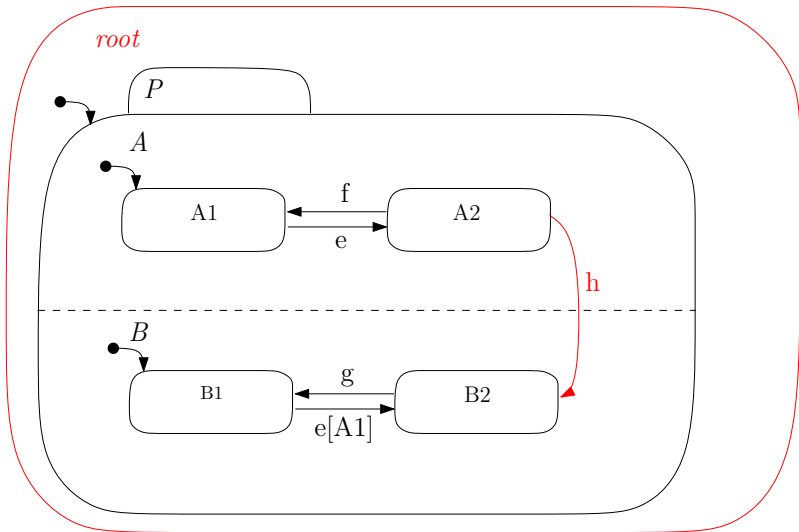
Configuration w.r.t. *root*?

- $\{root, P, A, A2, B, B2\}$ ✓

Semantical Puzzel: Scope



Semantical Puzzel: Scope



Statecharts

A statechart is a tuple (\mathcal{S}, E, Tr) such that the hierarchy \mathcal{S} is well-defined and the set of transitions Tr is well-defined.

Outline

1 Recap

2 Micro-Steps

3 Macro-Steps

4 Other Features

5 UML State Diagrams

6 Conclusions

Enabledness

A **situation** is a pair $(conf, Ev)$ of configuration $conf \subseteq S$ w.r.t. root and events $Ev \subseteq E$.

Enabledness

A **situation** is a pair $(conf, Ev)$ of configuration $conf \subseteq S$ w.r.t. root and events $Ev \subseteq E$.

Transition $tr = (S_0, a[c]/e, S_1)$ is **enabled** at situation $(conf, Ev)$ when

- 1 $S_0 \subseteq conf$,
- 2 $a \subseteq Ev$ and
- 3 $c \subseteq conf$.

Enabledness

A **situation** is a pair $(conf, Ev)$ of configuration $conf \subseteq S$ w.r.t. root and events $Ev \subseteq E$.

Transition $tr = (S_0, a[c]/e, S_1)$ is **enabled** at situation $(conf, Ev)$ when

- 1 $S_0 \subseteq conf$,
- 2 $a \subseteq Ev$ and
- 3 $c \subseteq conf$.

Given a statechart $SC = (S, E, Tr)$ and a situation st , the set of all **enabled** transitions is denoted by $enabled(st)$.

Unit Exit State

Idea: The child of scope containing all source states.

Unique exit state of $tr = (S_0, l, S_1)$ is a state $UExit(tr) \in child(scope(tr))$ such that $UExit(tr) \in Anc(S_0)$.

Unit Exit State

Idea: The child of scope containing all source states.

Unique exit state of $tr = (S_0, l, S_1)$ is a state $UExit(tr) \in child(scope(tr))$ such that $UExit(tr) \in Anc(S_0)$.

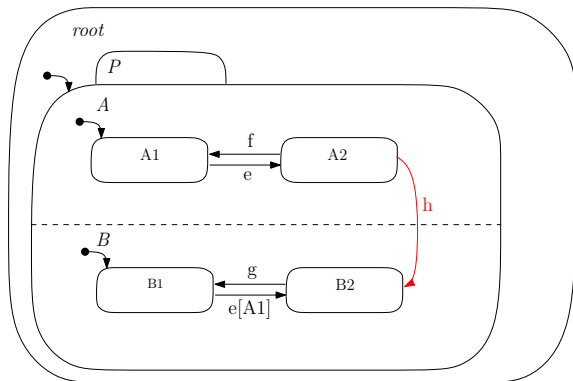
Exit Set

Idea: The set of states left due to the transition.

Given a configuration $conf$ w.r.t. root, $Ext(tr) = conf \cap desc(UExit(tr))$.

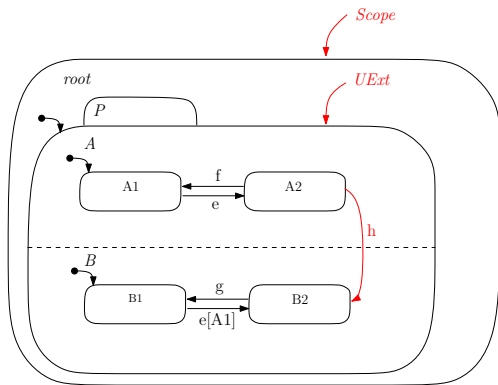
Semantical Puzzel: Exit Set

Given $conf = \{root, P, A, A2, B, B1\}$



Semantical Puzzel: Exit Set

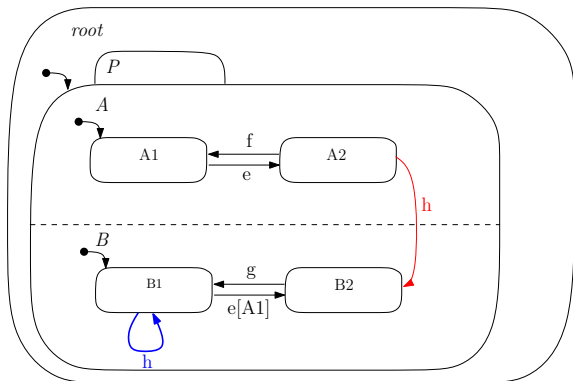
Given $conf = \{root, P, A, A2, B, B1\}$



$Ext(h) = \{P, A, A2, B, B1\}$

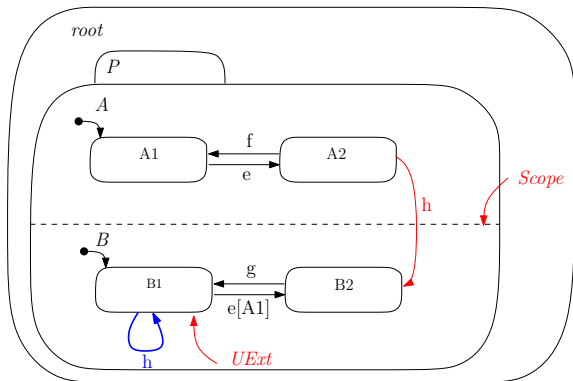
Semantical Puzzel: Exit Set

Given $conf = \{root, P, A, A2, B, B1\}$



Semantical Puzzel: Exit Set

Given $conf = \{root, P, A, A2, B, B1\}$



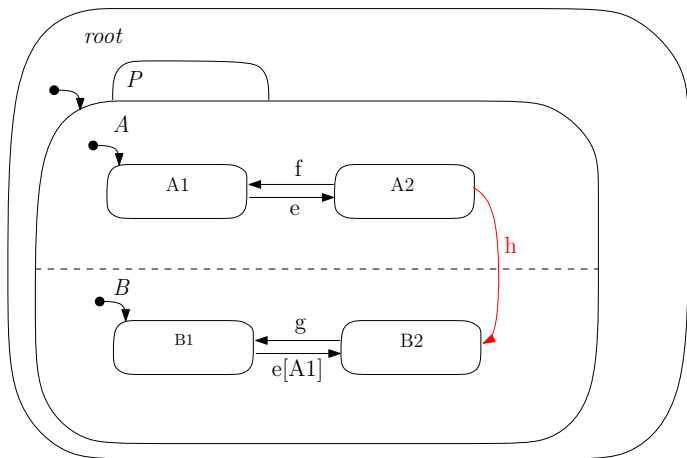
$Ext(h) = \{B1\}$

Enter Set

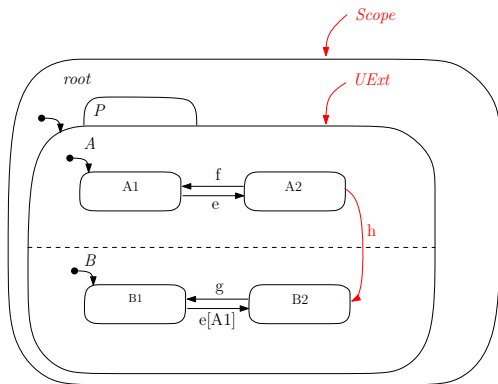
Unique enter state of $tr = (S_0, l, S_1)$ is a state $UEnt(tr) \in child(scope(tr))$ and $UEnt(tr) \in Anc(S_1)$.

Enter set of tr is a **configuration** $Ent(tr)$ w.r.t. $UEnt(tr)$ such that $S_1 \subseteq Ent(tr)$ and for all $s \in Ent(tr) \setminus Anc(S_1)$, it holds that $default(s)$.

Semantical Puzzel: Enter Set



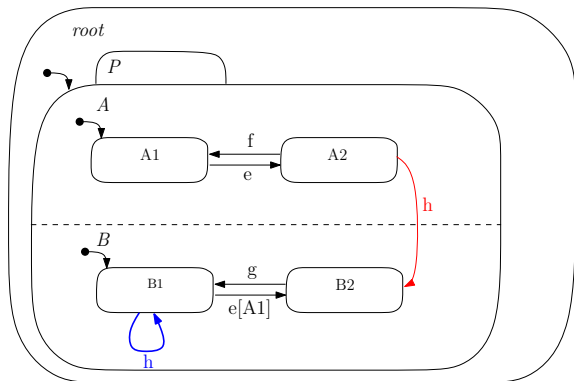
Semantical Puzzel: Enter Set



$$Ent(h) = \{P, A, A1, B, B2\}$$

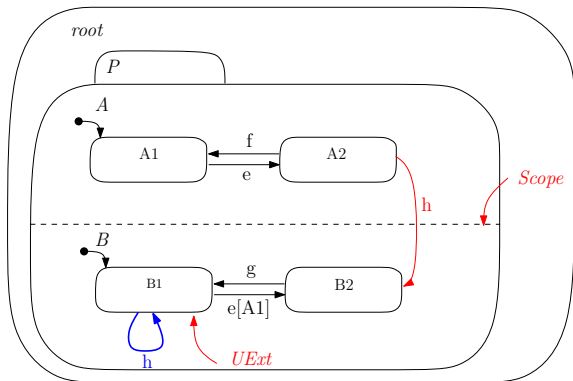
Semantical Puzzel: Enter Set

Given $conf = \{root, P, A, A2, B, B1\}$



Semantical Puzzel: Enter Set

Given $conf = \{root, P, A, A2, B, B1\}$



$Ent(h) = \{B1\}$

Consistency

Consistent transitions

Idea: transitions that can be taken simultaneously are consistent.

Consistent transitions

Idea: transitions that can be taken simultaneously are consistent.

Given a configuration $conf$, two transitions tr and tr' are **consistent** when $Ext(tr) \cap Ext(tr') = \emptyset$.

Consistent transitions

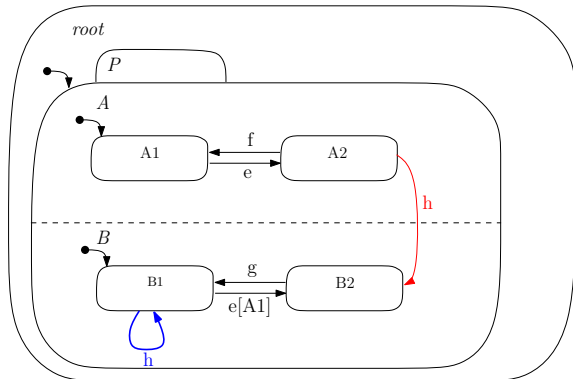
Idea: transitions that can be taken simultaneously are consistent.

Given a configuration $conf$, two transitions tr and tr' are **consistent** when $Ext(tr) \cap Ext(tr') = \emptyset$.

A set trs of transitions is **pairwise consistent** when for all $tr, tr' \in trs$, if $tr \neq tr'$ then tr and tr' are consistent.

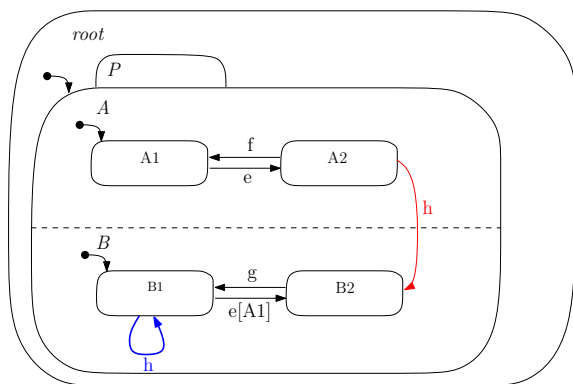
Semantical Puzzel: Consistency

Given $conf = \{root, P, A, A2, B, B1\}$



Semantical Puzzel: Consistency

Given $conf = \{root, P, A, A2, B, B1\}$



$Ext(h) = \{B1\}$

$Ext(h) = \{P, A, A2, B, B1\}$

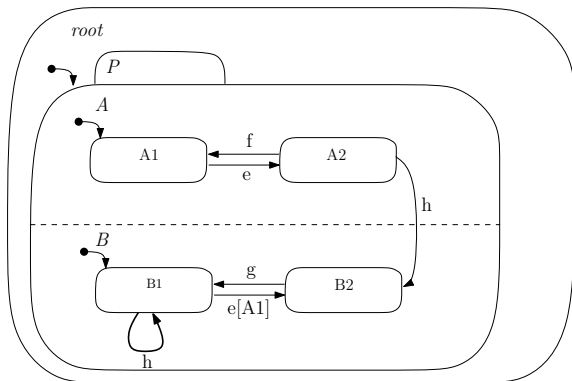
Maximal Micro-steps

Given a statechart SC and situation st , a maximal set of micro-steps $MMS(st) \subseteq enabled(st)$ is a pairwise-consistent set of transitions such that $\forall ms \in enabled(st) \setminus MMS(st), \{ms\} \cup MMS(st)$ is not pairwise consistent.

Note that $MMS(st)$ is not necessarily unique.

Semantical Puzzel: MMS

Given $st = (\{root, P, A, A2, B, B1\}, \{h, f\})$



Cause and Effect of a Micro-Step

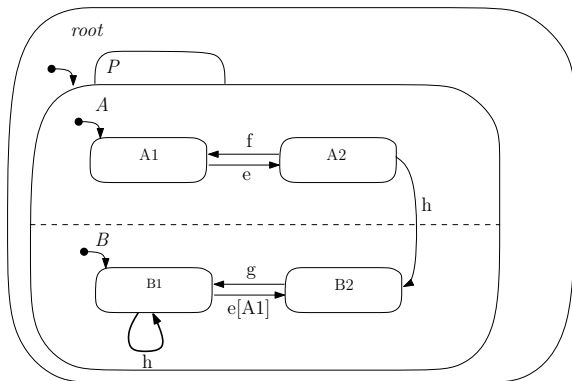
Given a situation $st = (conf, e)$ and $MMS(st) = \{tr_0, tr_1, \dots, tr_n\}$, where $tr_i = (S_i, T_i, C_i, G_i, S'_i)$, we write

$(conf, e) \xrightarrow{T_0 \cup \dots \cup T_n / G_0 \cup \dots \cup G_n} (conf', e')$ when

- 1 $conf' = conf \setminus (Ext(tr_0) \cup Ext(tr_1) \cup \dots \cup Ext(tr_n)) \cup (Ent(tr_0) \cup \dots \cup Ent(tr_n))$
and
- 2 $e' = G_0 \cup \dots \cup G_n$.

Semantical Puzzel: MMS

Given $st = (\{root, P, A, A2, B, B1\}, \{h, f\})$



Outline

1 Recap

2 Micro-Steps

3 Macro-Steps

4 Other Features

5 UML State Diagrams

6 Conclusions

LTS of Statechart

LTS of a statechart is a 5-tuple $(s_i, S_e, S_{sc}, \rightarrow_e, \rightarrow_{sc})$ where

- 1 the initial situation $s_i = (conf_0, \{\}) \in S_e$ where
 $\forall s : conf_0 \setminus \{root\} \bullet default(s);$

LTS of Statechart

LTS of a statechart is a 5-tuple $(s_i, S_e, S_{sc}, \rightarrow_e, \rightarrow_{sc})$ where

- 1 the initial situation $s_i = (conf_0, \{\}) \in S_e$ where
 $\forall s : conf_0 \setminus \{root\} \bullet default(s)$;
- 2 S_e is the set of **environment** situations;

LTS of Statechart

LTS of a statechart is a 5-tuple $(s_i, S_e, S_{sc}, \rightarrow_e, \rightarrow_{sc})$ where

- 1 the initial situation $s_i = (conf_0, \{\}) \in S_e$ where
 $\forall s : conf_0 \setminus \{root\} \bullet default(s)$;
- 2 S_e is the set of **environment** situations;
- 3 S_{sc} is a set of **statechart** situations;

LTS of Statechart

LTS of a statechart is a 5-tuple $(s_i, S_e, S_{sc}, \rightarrow_e, \rightarrow_{sc})$ where

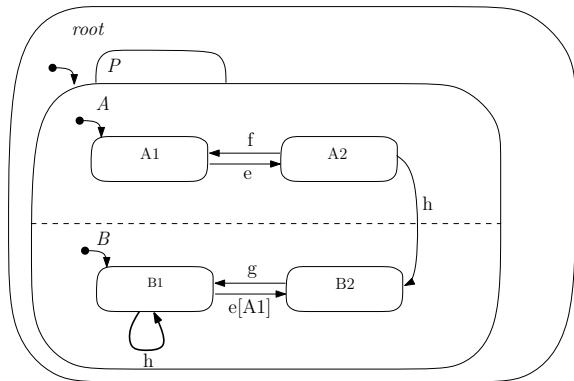
- 1 the initial situation $s_i = (conf_0, \{\}) \in S_e$ where
 $\forall s : conf_0 \setminus \{root\} \bullet default(s)$;
- 2 S_e is the set of **environment** situations;
- 3 S_{sc} is a set of **statechart** situations;
- 4 $\rightarrow_e \subseteq S_e \times (\mathbb{P} E) \times S_{sc}$: transitions by which the environment **adds triggers**; $((conf, ev), ev'', (conf', ev')) \in \rightarrow_e$ iff $conf = conf'$ and $ev' = ev \cup ev''$.

LTS of Statechart

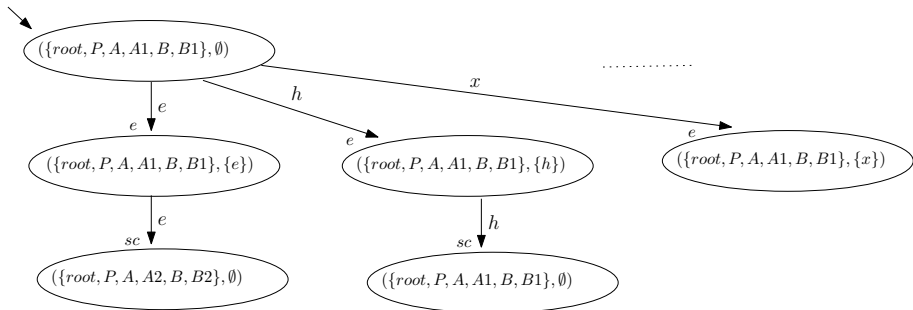
LTS of a statechart is a 5-tuple $(s_i, S_e, S_{sc}, \rightarrow_e, \rightarrow_{sc})$ where

- 1 the initial situation $s_i = (conf_0, \{\}) \in S_e$ where
 $\forall s : conf_0 \setminus \{root\} \bullet default(s)$;
- 2 S_e is the set of **environment** situations;
- 3 S_{sc} is a set of **statechart** situations;
- 4 $\rightarrow_e \subseteq S_e \times (\mathbb{P} E) \times S_{sc}$: transitions by which the environment **adds triggers**; $((conf, ev), ev'', (conf', ev')) \in \rightarrow_e$ iff $conf = conf'$ and $ev' = ev \cup ev''$.
- 5 $\rightarrow_{sc} \subseteq S_{sc} \times (\mathbb{P} E \times \mathbb{P} E) \times S_e$: transitions by which the **statechart reacts**; $((conf, ev), (ev_0, ev_1), (conf', ev')) \in \rightarrow_{sc}$ iff $(conf, ev) \xrightarrow{ev_0/ev_1} (conf', ev')$.

Semantical Puzzel: LTS (Partial)



Semantical Puzzel: LTS (Partial)



Outline

- 1 Recap
- 2 Micro-Steps
- 3 Macro-Steps
- 4 Other Features**
- 5 UML State Diagrams
- 6 Conclusions

- defined locally
- used in the condition of transitions
- changed by the triggered events
- semantically: an orthogonal statechart storing and updating the values

Other Connectors

- Junction, condition and switch connectors
- more concise specification of “similar” transitions
- semantically: a number of transitions

Additional Features

- Priority among transitions
- Time
- Negative events and event expressions
- More complex conditions

Outline

- 1 Recap
- 2 Micro-Steps
- 3 Macro-Steps
- 4 Other Features
- 5 UML State Diagrams**
- 6 Conclusions

UML 2 State Diagrams

- Behavior State Diagrams (basically what you have seen here)
- Protocol State Diagrams (specialization of Behavior SDs, pre- and post-conditions on labels)

UML State Diagrams

States

- Basic
- Regions
- Orthogonal states
- Composite states

State Actions

- Entry
- Do
- Exit

Pseudo-States

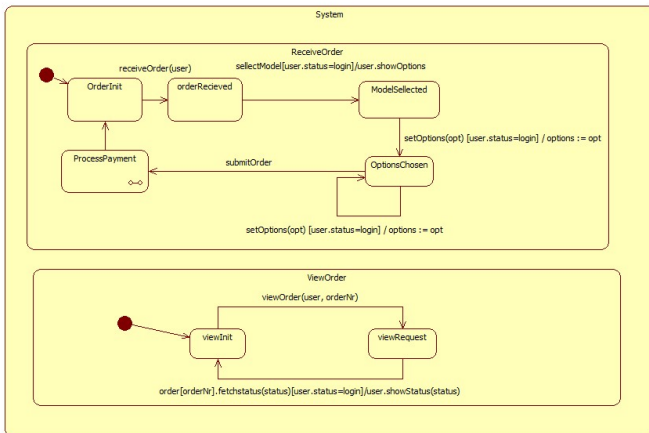
- Initial, final
- (deep) history
- join and fork
- decision and choice

State Diagrams in the Project

Per Class

- Specify a state diagram, break into multiple diagrams and give references if necessary
- Inherited classes are included as orthogonal states (children of an and-state)
- For each non-trivial method, add the state transitions realizing its functionality
- Assume attributes / associations

State Diagrams in the Project



Outline

- 1 Recap
- 2 Micro-Steps
- 3 Macro-Steps
- 4 Other Features
- 5 UML State Diagrams
- 6 Conclusions**

Features

- FSM-based model with **clustering**, **orthogonality** and communication
- an **integrated** model of **behavior**
(e.g., behavior of an object, or a composition of objects)
- rigorous **semantics**
(suitable for formal reasoning)
- features such as history connectors and data

Free Tools

- Yakindu Eclipse Plugin
- Xholon
- Statecharts Virtual Machine (SVM)

Commercial Tools

- StateMate by i-Logix (Telelogic)
- VisualState by IAR Systems
- Many UML-based tools: Artisan Studio, IBM Rhapsody,...

Papers

- E. Mikk, Y. Lakhnch, C. Paterson, M. Siegel, On Formal Semantics of Statecharts as Supported by Statemate. Proc. BCS-FACS Northern Formal Methods Workshop, 1997.
- D. Harel. Statecharts, a visual formalism for complex systems. *Science of Computer Programming*, 8(3):231–274, 1987.
- D. Harel, Statecharts in the making: a personal account. Proc. of ACM SIGPLAN HOPL-III, 2007.