Content

- Reflective Descriptions of Behaviour
  - Interactions
  - A Brief History of Sequence Diagrams

- Live Sequence Charts
  - Abstract Syntax, Well-Formedness
  - Semantics
    - TBA Construction for LSC Body
      - Cuts, Firedsets
      - Signal / Attribute Expressions
      - Loop / Progress Conditions
    - Excursion: Büchi Automata
    - Language of a Model
    - Full LSCs
      - Existential and Universal
      - Pre-Charts
      - Forbidden Scenarios
    - LSCs and Tests

---

LSC: not so simple can be concise
Sem: not so simple
Model: may be small

HS: simple/short
CS: complex/short

CH: large
ASM: simple/may get large

The Plan

- Thu, 19. 1.: Live Sequence Charts I
  Firstly: State-Machines Rest, Code Generation
- Tue, 24. 1.: Live Sequence Charts II
  - Thu, 26. 1.: Live Sequence Charts III
  - Tue, 31. 1.: Tutorial 7
  - Thu, 2. 2.: Model Based/Driven SW Engineering
  - Mon, 6. 2.: Inheritance
  - Tue, 7. 2.: Meta-Modelling + Questions

February, 17th: The Exam.
Constructive Behavioural Modelling in UML: Discussion
**Semantic Variation Points**

**Pessimistic view:** There are too many...

- For instance,
  - allow **absence of initial pseudo-states**
    object may then “be” in enclosing state without being in any substate;
or assume one of the children states non-deterministically
  - (implicitly) **enforce determinism**, e.g.
    by considering the order in which things have been added to the CASE tool's repository,
or some graphical order (left to right, top to bottom)
  - allow **true concurrency**
  - etc. etc.

**Exercise:** Search the standard for “semantical variation point”.

- **Crane and Dingel (2007),** e.g., provide an in-depth comparison of Statemate, UML, and Rhapsody state
  machines – the bottom line is:
    - **the intersection is not empty** (i.e. some diagrams mean the same to all three communities)
    - **none is the subset of another** (i.e. each pair of communities has diagrams meaning different things)

**Optimistic view:**

- tools exist with **complete and consistent** code generation.
- good modelling-guidelines can contribute to **avoiding misunderstandings**.
Reflective Descriptions of Behaviour
Constructive vs. Reflective Descriptions

Harel (1997) proposes to distinguish constructive and reflective descriptions:

- A constructive description tells us how things are computed:
  
  “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 reflective description tells us what shall (or shall not) be computed:
  
  “Other languages are **reflective** or **assertive**, and can be used by the system modeler 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.”

**Note:** No sharp boundaries! (Would be too easy.)
Interactions as Reflective Description

In UML, reflective (temporal) descriptions are subsumed by interactions. A UML model $\mathcal{M} = (\mathcal{CD}, \mathcal{IM}, \mathcal{OD}, \mathcal{I})$ has a set of interactions $\mathcal{I}$.

An interaction $\mathcal{I} \in \mathcal{I}$ can be (OMG claim: equivalently) diagrammed as
- communication diagram (formerly known as collaboration diagram),
- timing diagram, or
- sequence diagram.
Interactions as Reflective Description

- In UML, reflective (temporal) descriptions are subsumed by interactions. A UML model $\mathcal{M} = (\mathcal{C} \mathcal{D}, \mathcal{I} \mathcal{M}, \mathcal{O} \mathcal{D}, \mathcal{I})$ has a set of interactions $\mathcal{I}$.
- An interaction $\mathcal{I} \in \mathcal{I}$ can be (OMG claim: equivalently) diagrammed as
  - communication diagram (formerly known as collaboration diagram),
  - timing diagram, or
  - sequence diagram.

---

Figure 14.27 - Communication diagram

Figure 14.30 - Communication diagram

Figure 14.28 - Interaction Overview Diagram representing a High Level Interaction diagram

Figure 14.31 - Timing Diagram with more than one Lifeline and with Messages

Figure 14.26 - Sequence Diagram with time and timing concepts

Figure 14.31 - Sequence Diagram with time and timing concepts

(OMG, 2007, 515)
**Why Sequence Diagrams?**

**Most Prominent:** Sequence Diagrams – with **long history**:

- **Message Sequence Charts**, standardized by the ITU in different versions, often accused to lack a formal semantics.
- **Sequence Diagrams** of UML 1.x

Most severe **drawbacks** of these formalisms:

- unclear **interpretation**: example scenario or invariant?
- unclear **activation**: what triggers the requirement?
- unclear **progress** requirement: must all messages be observed?
- **conditions** merely comments
- no means to express **forbidden scenarios**
• SDs of UML 2.x address some issues, yet the standard exhibits unclarities and even contradictions Harel and Maoz (2007); Störrle (2003)

• For the lecture, we consider Live Sequence Charts (LSCs) Damm and Harel (2001); Klose (2003); Harel and Marelly (2003), who have a common fragment with UML 2.x SDs Harel and Maoz (2007)

• Modelling guideline: stick to that fragment.
\[ M = (\Sigma_{\mathcal{F}}, A_{\mathcal{F}}, \rightarrow_{SM}) \]

\[ \pi = (\sigma_0, \varepsilon_0) \frac{(\text{cons}_0, \text{Snd}_0)}{u_0} (\sigma_1, \varepsilon_1) \cdots \]

\[ w_\pi = (\sigma_i, \text{cons}_i, \text{Snd}_i)_{i \in \mathbb{N}} \]

\[ \mathcal{G} = (N, E, f) \]

\[ \varphi \in \text{OCL} \]

\[ \mathcal{L} = (\mathcal{T}, \mathcal{C}, V, \text{atr}), SM \]

\[ \varphi \in \text{OCL} \]

\[ B = (Q_{SD}, q_0, A_{\mathcal{F}}, \rightarrow_{SD}, F_{SD}) \]
Live Sequence Charts — Syntax
\[ v = 0 \]
LSC Body Building Blocks

- instance line head
- logical variable
- simultaneous region
- (cold) line segment
- (hot) line segment
- (cold) local invariant
- exclusive
- inclusive
- instantaneous message
- (hot) condition
- asynchronous message
- coregion
- life line / instance line
LSC: buy water
AC: true
AM: invariant I: strict

User
CoinValidator
\(cp: \text{ChoicePanel}\)
Dispenser

\(~(C50 \lor E1 \lor pSOFT! \lor pTEA! \lor pFILLUP!)\)

\(~(dSOFT! \lor dTEA!)\)

\(cp \rightarrow \text{water\_in\_stock}\)

\(dWATER\)

\(OK\)

\(\text{environment}\)

LSC name
activation condition
activation mode
(invariant or initial)
interpretation (strict or permissive)
 logical variable
class name
pre-chart
(hot) main chart
**Definition.** [LSC Body]

An LSC body over signature \( \mathcal{S} = (\mathcal{T}, \mathcal{C}, V, atr, \mathcal{E}) \) is a tuple

\[
((L, \preceq, \sim), \mathcal{I}, \text{Msg}, \text{Cond}, \text{LocInv}, \Theta)
\]

where

- \( L \) is a finite, non-empty set of locations with
  - a partial order \( \preceq \subseteq L \times L \),
  - a symmetric simultaneity relation \( \sim \subseteq L \times L \) disjoint with \( \preceq \), i.e. \( \preceq \cap \sim = \emptyset \),
- \( \mathcal{I} = \{I_1, \ldots, I_n\} \) is a partitioning of \( L \); elements of \( \mathcal{I} \) are called instance line,
- \( \text{Msg} \subseteq L \times \mathcal{E} \times L \) is a set of messages with \( (l, E, l') \in \text{Msg} \) only if \( (l, l') \in \prec \cup \sim \); message \( (l, E, l') \) is called instantaneous iff \( l \sim l' \) and asynchronous otherwise,
- \( \text{Cond} \subseteq (2^L \setminus \emptyset) \times \text{Expr}_\mathcal{S} \) is a set of conditions with \( (L, \phi) \in \text{Cond} \) only if \( l \sim l' \) for all \( l \neq l' \in L \),
- \( \text{LocInv} \subseteq L \times \{\circ, \bullet\} \times \text{Expr}_\mathcal{S} \times L \times \{\circ, \bullet\} \) is a set of local invariants with \( (l, \iota, \phi, l', \iota') \in \text{LocInv} \) only if \( l \prec l' \), \( \circ \): exclusive, \( \bullet \): inclusive,
- \( \Theta : L \cup \text{Msg} \cup \text{Cond} \cup \text{LocInv} \rightarrow \{\text{hot, cold}\} \) assigns to each location and each element a temperature.
locations $L$,
• $\leq \subseteq L \times L$, $\sim \subseteq L \times L$
• $\mathcal{I} = \{I_1, \ldots, I_n\}$,
• $\text{Msg} \subseteq L \times \mathcal{E} \times L$,
• $\text{Cond} \subseteq (2^L \setminus \emptyset) \times \text{Expr}_\mathcal{S}$
• $\text{LocInv} \subseteq L \times \{\circ, \bullet\} \times \text{Expr}_\mathcal{S} \times L \times \{\circ, \bullet\}$,
• $\Theta : L \cup \text{Msg} \cup \text{Cond} \cup \text{LocInv} \rightarrow \{\text{hot, cold}\}$. 
locations $L$, $\leq \subseteq L \times L$, $\sim \subseteq L \times L$

$\mathcal{I} = \{I_1, \ldots, I_n\}$,

$\text{Msg} \subseteq L \times \mathcal{E} \times L$,

$\text{Cond} \subseteq (2^L \setminus \emptyset) \times \text{Expr}$

$\text{LocInv} \subseteq L \times \{\circ, \bullet\} \times \text{Expr} \times L \times \{\circ, \bullet\}$,

$\Theta : L \cup \text{Msg} \cup \text{Cond} \cup \text{LocInv} \rightarrow \{\text{hot, cold}\}$

\begin{align*}
L & = \{l_{2,0}, \ldots, l_{1,1}, l_{1,0}, \ldots, l_{2,3}, l_{3,0}, \ldots, l_{3,2}\} \\
\mathcal{I} & = \{(l_{1,0}, l_{1,1}), (l_{1,1}, l_{1,2}), (l_{1,2}, l_{1,3}), (l_{1,3}, l_{1,4}), \ldots, (l_{1,4}, l_{2,1}), (l_{2,2}, l_{2,3}), \ldots\}
\end{align*}

\begin{align*}
\leq & = \preceq^* \\
\sim & = \{(l_{2,2}, l_{3,1})\}
\end{align*}

\begin{align*}
\text{Msg} & = \{(l_{1,1}, A, l_{2,1})\} \\
\text{Cond} & = \{(\{l_{2,2}\}, \star > 3)\} \\
\text{LocInv} & = \{(l_{1,1}, 0, \sqrt{v = 0}, l_{1,2}, \bullet)\}
\end{align*}

$\Theta = \{M \mapsto \text{hot}, C' \mapsto \text{hot}, I \mapsto \text{cold}, l_{3,0} \mapsto \text{cold}, l_{2,1} \mapsto \text{hot}, \ldots\}$
• locations $L$,
• $\preceq \subseteq L \times L$, $\sim \subseteq L \times L$
• $\mathcal{I} = \{I_1, \ldots, I_n\}$,
• $\text{Msg} \subseteq L \times \mathcal{E} \times L$,
• $\text{Cond} \subseteq (2^L \setminus \emptyset) \times \text{Expr} \mathcal{S}$
• $\text{LocInv} \subseteq L \times \{\circ, \bullet\} \times \text{Expr} \mathcal{S} \times L \times \{\circ, \bullet\}$,
• $\Theta : L \cup \text{Msg} \cup \text{Cond} \cup \text{LocInv} \to \{\text{hot, cold}\}$. 

---

**Diagram:**

- Nodes labeled with 'NO:', 'C:', and 'D:'
- Edges connecting the nodes with arrows indicating flow or transition.
**Well-Formedness**

**Bondedness/no floating conditions**: (could be relaxed a little if we wanted to)

- For each location \( l \in L \), **if** \( l \) is the location of
  - a **condition**, i.e. \( \exists (L, \phi) \in \text{Cond} : l \in L \), or
  - a **local invariant**, i.e. \( \exists (l_1, \nu_1, \phi, l_2, \nu_2) \in \text{LocInv} : l \in \{l_1, l_2\} \),

  then there is a location \( l' \) **simultaneous** to \( l \), i.e. \( l \sim l' \), which is the location of
  - an **instance head**, i.e. \( l' \) is minimal wrt. \( \preceq \), or
  - a **message**, i.e.

\[
\exists (l_1, E, l_2) \in \text{Msg} : l \in \{l_1, l_2\}.
\]

**Note**: if messages in a chart are **cyclic**, then there doesn’t exist a partial order (so such diagrams **don’t even have** an abstract syntax).
Live Sequence Charts — Semantics
Plan:

(i) Given an LSC \( \mathcal{L} \) with body

\[ \left( (L, \preceq, \sim), \mathcal{I}, \text{Msg}, \text{Cond}, \text{LocInv}, \Theta \right), \]

(ii) construct a TBA \( B_{\mathcal{L}} \), and

(iii) define language \( \mathcal{L}(\mathcal{L}) \) of \( \mathcal{L} \) in terms of \( \mathcal{L}(B_{\mathcal{L}}) \),

in particular taking activation condition and activation mode into account.

(iv) define language \( \mathcal{L}(\mathcal{M}) \) of a UML model.

- Then \( \mathcal{M} \models \mathcal{L} \) (universal) if and only if \( \mathcal{L}(\mathcal{M}) \subseteq \mathcal{L}(\mathcal{L}) \).
- And \( \mathcal{M} \models \mathcal{L} \) (existential) if and only if \( \mathcal{L}(\mathcal{M}) \cap \mathcal{L}(\mathcal{L}) \neq \emptyset \).
Live Sequence Charts — TBA Construction
Definition.
Let \(((L, \preceq, \sim), \mathcal{I}, \text{Msg}, \text{Cond}, \text{Locinv}, \Theta)\) be an LSC body.
A non-empty set \(\emptyset \neq C \subseteq L\) is called a **cut** of the LSC body iff

- it is **downward closed**, i.e. \(\forall l, l' \bullet l' \in C \land l \preceq l' \implies l \in C\),
- it is **closed under simultaneity**, i.e.
  \[
  \forall l, l' \bullet l' \in C \land l \sim l' \implies l \in C, \text{ and}
  \]
- it comprises at least **one location per instance line**, i.e.
  \[
  \forall i \in I \exists l \in C \bullet i_l = i.
  \]
**Definition.**
Let \(((L, \preceq, \sim), \mathcal{I}, \text{Msg}, \text{Cond}, \text{LocInv}, \Theta)\) be an LSC body.

A non-empty set \(\emptyset \neq C \subseteq L\) is called a **cut** of the LSC body iff

- it is **downward closed**, i.e. \(\forall l, l' \bullet l' \in C \land l \preceq l' \implies l \in C\),
- it is **closed under simultaneity**, i.e.
  \[
  \forall l, l' \bullet l' \in C \land l \sim l' \implies l \in C, \text{ and}
  \]
- it comprises at least **one location per instance line**, i.e.
  \[
  \forall i \in I \exists l \in C \bullet i_l = i.
  \]

The **temperature function** is extended to cuts as follows:

\[
\Theta(C) = \begin{cases} 
  \text{hot} & \text{if } \exists l \in C \bullet (\nexists l' \in C \bullet l \prec l') \land \Theta(l) = \text{hot} \\
  \text{cold} & \text{otherwise}
\end{cases}
\]

that is, \(C\) is **hot** if and only if at least one of its maximal elements is hot.
\[ \emptyset \neq C \subseteq L \text{ – downward closed – simultaneity closed – at least one loc. per instance line} \]
\( \emptyset \neq C \subseteq L \) – downward closed – simultaneity closed – at least one loc. per instance line
Cut Examples

\[ \emptyset \neq C \subseteq L \text{ – downward closed – simultaneity closed – at least one loc. per instance line} \]
\( \emptyset \neq C \subseteq L \) – downward closed – simultaneity closed – at least one loc. per instance line
$\emptyset \neq C \subseteq L$ – downward closed – simultaneity closed – at least one loc. per instance line
∅ ≠ C ⊆ L – downward closed – simultaneity closed – at least one loc. per instance line
$\emptyset \neq C \subseteq L$ – downward closed – simultaneity closed – at least one loc. per instance line
∅ ≠ C ⊆ L – downward closed – simultaneity closed – at least one loc. per instance line
A Successor Relation on Cuts

The partial order “\( \preceq \)” and the simultaneity relation “\( \sim \)” of locations induce a direct successor relation on cuts of an LSC body as follows:

**Definition.**
Let \( C \subseteq L \) bet a cut of LSC body \(((L, \preceq, \sim), \mathcal{I}, \text{Msg}, \text{Cond}, \text{LocInv}, \Theta)\).

A set \( \emptyset \neq F \subseteq L \) of locations is called fired-set \( F \) of cut \( C \) if and only if

- \( C \cap F = \emptyset \) and \( C \cup F \) is a cut, i.e. \( F \) is closed under simultaneity,
- all locations in \( F \) are direct \( \prec \)-successors of the front of \( C \), i.e.
  \[
  \forall l \in F \ \exists l' \in C : l' \prec l \land (\nexists l'' \in C : l' \prec l'' \prec l),
  \]
- locations in \( F \), that lie on the same instance line, are pairwise unordered, i.e.
  \[
  \forall l \neq l' \in F : (\exists I \in \mathcal{I} : \{l, l'\} \subseteq I) \implies l \npreceq l' \land l' \npreceq l,
  \]
- for each asynchronous (!) message reception in \( F \), the corresponding sending is already in \( C \),
  \[
  \forall (l, E, l') \in \text{Msg} : l' \in F \implies l \in C.
  \]

The cut \( C' = C \cup F \) is called direct successor of \( C \) via \( F \), denoted by \( C \sim_F C' \).
$C \cap F = \emptyset$ – $C \cup F$ is a cut – only direct $\prec$-successors – same instance line on front pairwise unordered – sending of asynchronous reception already in
$C \cap F = \emptyset$ – $C \cup F$ is a cut – only direct $\prec$-successors – same instance line on front pairwise unordered – sending of asynchronous reception already in
Language of LSC Body: Example
Language of LSC Body: Example
The TBA $B_L$ of LSC $L$ over $\Phi$ and $E$ is $(Expr_B(X), X, Q, q_{ini}, \rightarrow, Q_F)$ with

- $Q$ is the set of cuts of $L$, $q_{ini}$ is the instance heads cut,
- $Expr_B(X) = Expr_\mathcal{P}(\mathcal{E}, X)$ (for considered signature $\mathcal{P}$),
- $\rightarrow$ consists of loops, progress transitions (by $\rightsquigarrow_F$), and legal exits (cold cond./local inv.),
- $Q_F = \{ C \in Q \mid \Theta(C) = \text{cold} \lor C = L \}$ is the set of cold cuts and the maximal cut.
Signal and Attribute Expressions

- Let $\mathcal{S} = (\mathcal{T}, \mathcal{C}, V, atr, \mathcal{E})$ be a signature and $X$ a set of logical variables,

- The signal and attribute expressions $Expr_\mathcal{S}(\mathcal{E}, X)$ are defined by the grammar:

$$\psi ::= true \mid \psi \mid E^!_{x,y} \mid E^?_{x,y} \mid \neg \psi \mid \psi_1 \lor \psi_2,$$

where $expr : Bool \in Expr_\mathcal{S}, E \in \mathcal{E}, x, y \in X$ (or keyword $env$).

- We use

$$\mathcal{E}!(X) := \{ E^!_{x,y}, E^?_{x,y} \mid E \in \mathcal{E}, x, y \in X \}$$

to denote the set of event expressions over $\mathcal{E}$ and $X$. 
Recall: The TBA $B(L)$ of LSC $L$ is $(Expr_B(X), X, Q, q_{ini}, \rightarrow, Q_F)$ with

- $Q$ is the set of cuts of $L$, $q_{ini}$ is the instance heads cut,
- $Expr_B = \Phi \cup E!?(X)$,
- $\rightarrow$ consists of loops, progress transitions (from $\leadsto F$), and legal exits (cold cond./local inv.),
- $F = \{ C \in Q \mid \Theta(C) = \text{cold} \lor C = L \}$ is the set of cold cuts.
Recall: The TBA $\mathcal{B}(\mathcal{L})$ of LSC $\mathcal{L}$ is $(\text{Expr}_B(X), X, Q, q_{ini}, \rightarrow, Q_F)$ with

- $Q$ is the set of cuts of $\mathcal{L}$, $q_{ini}$ is the instance heads cut,
- $\text{Expr}_B = \Phi \cup \mathcal{E}_{?}(X)$,
- $\rightarrow$ consists of loops, progress transitions (from $\rightsquigarrow_F$), and legal exits (cold cond./local inv.),
- $F = \{ C \in Q \mid \Theta(C) = \text{cold} \lor C = L \}$ is the set of cold cuts.

So in the following, we “only” need to construct the transitions’ labels:

$\rightarrow = \{(q, , q) \mid q \in Q\} \cup \{(q, , q') \mid q \rightsquigarrow_F q'\} \cup \{(q, , L) \mid q \in Q\}$
Recall: The TBA $\mathcal{B}(\mathcal{L})$ of LSC $\mathcal{L}$ is $(Expr_\mathcal{B}(X), X, Q, q_{ini}, \rightarrow, Q_F)$ with

- $Q$ is the set of cuts of $\mathcal{L}$, $q_{ini}$ is the instance heads cut,
- $Expr_\mathcal{B} = \emptyset \cup E_{1?}(X)$,
- $\rightarrow$ consists of loops, progress transitions (from $\rightsquigarrow_F$), and legal exits (cold cond./local inv.),
- $F = \{ C \in Q \mid \Theta(C) = \text{cold} \lor C = L \}$ is the set of cold cuts.

So in the following, we “only” need to construct the transitions’ labels:

$$\rightarrow = \{(q, \psi_{\text{loop}}(q), q) \mid q \in Q\} \cup \{(q, \psi_{\text{prog}}(q, q'), q') \mid q \rightsquigarrow_F q'\} \cup \{(q, \psi_{\text{exit}}(q), L) \mid q \in Q\}$$
**Recall:** The TBA $\mathcal{B}(\mathcal{L})$ of LSC $\mathcal{L}$ is $(\text{Expr}_B(X), X, Q, q_{\text{ini}}, \rightarrow, Q_F)$ with

- $Q$ is the set of cuts of $\mathcal{L}$, $q_{\text{ini}}$ is the instance heads cut,
- $\text{Expr}_B = \Phi \cup \mathcal{E}_?^b(X)$,
- $\rightarrow$ consists of loops, progress transitions (from $\leadsto F$), and legal exits (cold cond./local inv.),
- $F = \{ C \in Q \mid \Theta(C) = \text{cold} \lor C = L \}$ is the set of cold cuts.

So in the following, we “only” need to construct the transitions’ labels:

$$
\rightarrow = \{(q, \psi_{\text{loop}}(q), q) \mid q \in Q\} \cup \{(q, \psi_{\text{prog}}(q, q'), q') \mid q \leadsto_F q'\} \cup \{(q, \psi_{\text{exit}}(q), L) \mid q \in Q\}
$$
“Only” construct the transitions’ labels:

\[\rightarrow = \{(q, \psi_{\text{loop}}(q), q) \mid q \in Q\} \cup \{(q, \psi_{\text{prog}}(q, q'), q') \mid q \xrightarrow{F} q'\} \cup \{(q, \psi_{\text{exit}}(q), L) \mid q \in Q\}\]

\[\psi_{\text{loop}}(q) = \psi_{\text{Msg}}(q) \land \psi_{\text{hot}}(q) \land \psi_{\text{loc inv}}(q)\]

\[\psi_{\text{exit}}(q) = (\psi_{\text{loop}}(q) \land \neg \psi_{\text{loc inv}}(q)) \lor \bigvee_{1 \leq i \leq n} (\psi_{\text{prog}}(q, q_i) \land \neg \psi_{\text{loc inv}}(q, q_i) \lor \neg \psi_{\text{cold}}(q, q_i))\]

\[\psi_{\text{prog}}(q, q_n) = \psi_{\text{Msg}}(q, q_n) \land \psi_{\text{cond}}(q, q_n) \land \psi_{\text{loc inv}, \bullet}(q, q_n) \land \psi_{\text{cold}}(q, q_n) \land \psi_{\text{cold}}(q, q_n)\]
Loop Condition

\[ \psi_{\text{loop}}(q) = \psi^{\text{Msg}}(q) \land \psi^{\text{LocInv}}_{\text{hot}}(q) \land \psi^{\text{LocInv}}_{\text{cold}}(q) \]

- \( \psi^{\text{Msg}}(q) = \neg \bigvee_{1 \leq i \leq n} \psi^{\text{Msg}}(q, q_i) \land (\text{strict} \implies \bigwedge_{\psi \in \text{Msg}(L)} \neg \psi) =: \psi_{\text{strict}}(q) \)

- \( \psi^{\text{LocInv}}_{\theta}(q) = \bigwedge_{\ell = (l, \nu, \phi, l', \nu') \in \text{LocInv}, \Theta(\ell) = \theta, \ell \text{ active at } q} \phi \)

A location \( l \) is called \textbf{front location} of cut \( C \) if and only if \( \nexists l' \in L \bullet l < l' \).

Local invariant \((l_0, \nu_0, \phi, l_1, \nu_1)\) is \textbf{active} at cut (!) \( q \) if and only if \( l_0 \preceq l < l_1 \) for some front location \( l \) of cut \( q \) or \( l_1 \in q \land \nu_1 = \bullet \).

- \( \text{Msg}(F) = \{ E^l_{x_l, x_l'} \mid (l, E, l') \in \text{Msg}, l \in F \} \cup \{ E^r_{x_l, x_l'} \mid (l, E, l') \in \text{Msg}, l' \in F \} \)

- \( x_l \in X \) is the logical variable associated with the instance line \( I \) which includes \( l \), i.e. \( l \in I \).

- \( \text{Msg}(F_1, \ldots, F_n) = \bigcup_{1 \leq i \leq n} \text{Msg}(F_i) \)
Progress Condition

$$\psi_{\text{prog}}(q, q_i) = \psi^\text{Msg}(q, q_n) \land \psi^\text{Cond}(q, q_n) \land \psi^\text{LocInv, •}(q_n)$$

- $$\psi^\text{Msg}(q, q_i) = \bigwedge_{\psi \in \text{Msg}(q_i \setminus q)} \psi \land \bigwedge_{j \neq i} \bigwedge_{\psi \in \text{Msg}(q_j \setminus q) \setminus \text{Msg}(q_i \setminus q)} \neg \psi \land \left(\text{strict} \implies \bigwedge_{\psi \in \text{Msg}(L) \setminus \text{Msg}(F_i)} \neg \psi\right) =: \psi_{\text{strict}}(q, q_i)$$

- $$\psi^\text{Cond}(q, q_i) = \bigwedge_{\gamma=(L, \phi) \in \text{Cond}, \Theta(\gamma)=\theta, L \cap (q_i \setminus q) \neq \emptyset} \phi$$

- $$\psi^\text{LocInv, •}(q, q_i) = \bigwedge_{\lambda=(l, \iota, \phi, l', \iota')} \in \text{LocInv}, \Theta(\lambda)=\theta, \lambda \bullet \text{-active at } q_i \phi$$

Local invariant $$(l_0, \iota_0, \phi, l_1, \iota_1)$$ is $\bullet$-active at q if and only if

- $$l_0 \prec l \prec l_1$$, or
- $$l = l_0 \land \iota_0 = \bullet$$, or
- $$l = l_1 \land \iota_1 = \bullet$$

for some front location l of cut (!) q.
Example

Using logical variables $x, y, z$ for the instances lines (from left to right).
\( \varphi \in \text{OCL} \)

\( \mathcal{I} = (\mathcal{F}, \mathcal{C}, \mathcal{V}, \text{atr}), \text{SM} \)

\( M = (\Sigma_{\mathcal{F}}, A_{\mathcal{F}}, \rightarrow_{\text{SM}}) \)

\( \pi = (\sigma_0, \epsilon_0) \xrightarrow{\text{cons}_0, \text{Snd}_0} (\sigma_1, \epsilon_1) \ldots \)

\( G = (N, E, f) \)

\( \mathcal{O} \mathcal{D} \)

\( \odot \mathcal{D} \)

\( \varphi \in \text{OCL} \)

\( \mathcal{I}, \text{SD} \)

\( B = (Q_{\text{SD}}, q_0, A_{\mathcal{F}}, \rightarrow_{\text{SD}}, F_{\text{SD}}) \)

\( w_{\pi} = ((\sigma_i, \text{cons}_i, \text{Snd}_i))_{i \in \mathbb{N}} \)
Interactions can be **reflective** descriptions of behaviour, i.e.
- describe **what** behaviour is (un)desired,
  without (yet) defining **how** to realise it.

One visual formalism for interactions: **Live Sequence Charts**
- locations in diagram **induce a partial order**, 
- instantaneous and asynchronous messages, 
- conditions and local invariants

The **meaning** of an LSC is defined using TBAs.
- **Cuts** become states of the automaton. 
- Locations induce a **partial order on cuts**. 
- Automaton-transitions and annotations correspond to a **successor relation** on cuts. 
- Annotations use **signal / attribute expressions**.

**Later:**
- TBA have **Büchi acceptance** (of infinite words (of a model)).
- **Full LSC semantics.**
- **Pre-Charts.**
Excursion: Büchi Automata
From Finite Automata to Symbolic Büchi Automata

\( A: \quad \Sigma = \{0, 1\} \)

\[ \begin{array}{c}
  q_1 \quad 0 \quad q_2 \\
  1
\end{array} \]

\( A_{\text{sym}}: \quad \Sigma = (\{x\} \to \mathbb{N}) \)

\[ \begin{array}{c}
  q_1 \quad \text{even}(x) \quad q_2 \\
  \text{odd}(x)
\end{array} \]

\( B: \quad \Sigma = \{0, 1\} \)

\[ \begin{array}{c}
  q_1 \quad 0 \quad q_2 \\
  1
\end{array} \]

\( B'_{\text{sym}}: \quad \Sigma = (\{x\} \to \mathbb{N}) \)

\[ \begin{array}{c}
  q_1 \quad \text{even}(x) \quad q_2 \\
  \text{odd}(x)
\end{array} \]

\( B_{\text{sym}}: \quad \Sigma = (\{x\} \to \mathbb{N}) \)

\[ \begin{array}{c}
  q_1 \quad 0 \\
  1 \quad 0
\end{array} \]
Definition. A **Symbolic Büchi Automaton** (TBA) is a tuple

\[ \mathcal{B} = (Expr_\mathcal{B}(X), X, Q, q_{ini}, \rightarrow, Q_F) \]

where

- \( X \) is a set of logical variables,
- \( Expr_\mathcal{B}(X) \) is a set of Boolean expressions over \( X \),
- \( Q \) is a finite set of **states**,
- \( q_{ini} \in Q \) is the initial state,
- \( \rightarrow \subseteq Q \times Expr_\mathcal{B}(X) \times Q \) is the **transition relation**. Transitions \((q, \psi, q')\) from \( q \) to \( q' \) are labelled with an expression \( \psi \in Expr_\mathcal{B}(X) \).
- \( Q_F \subseteq Q \) is the set of **fair** (or accepting) states.
Definition. Let $X$ be a set of logical variables and let $Expr_B(X)$ be a set of Boolean expressions over $X$.

A set $(\Sigma, \cdot \mid \cdot)$ is called an **alphabet** for $Expr_B(X)$ if and only if

- for each $\sigma \in \Sigma$,
  - for each expression $expr \in Expr_B$, and
  - for each valuation $\beta : X \rightarrow \mathcal{P}(X)$ of logical variables,

  either $\sigma \models_\beta expr$ or $\sigma \not\models_\beta expr$.

($\sigma$ satisfies (or does not satisfy) $expr$ under valuation $\beta$)

An **infinite sequence**

$$w = (\sigma_i)_{i \in \mathbb{N}_0} \in \Sigma^\omega$$

over $(\Sigma, \cdot \mid \cdot)$ is called **word** (for $Expr_B(X)$).
**Definition.** Let \( B = (Expr_B(X), X, Q, q_{ini}, \rightarrow, Q_F) \) be a TBA and
\[
w = \sigma_1, \sigma_2, \sigma_3, \ldots
\]
a word for \( Expr_B(X) \). An infinite sequence
\[
q = q_0, q_1, q_2, \ldots \in Q^\omega
\]
is called **run of** \( B \) **over** \( w \) **under valuation** \( \beta : X \rightarrow \mathcal{D}(X) \) if and only if

- \( q_0 = q_{ini} \),
- for each \( i \in \mathbb{N}_0 \) there is a transition
  \[
  (q_i, \psi_i, q_{i+1}) \in \rightarrow
  \]
such that \( \sigma_i \models_\beta \psi_i \).
**Definition.** Let $B = (\text{Expr}_B(X), X, Q, q_{\text{ini}}, \rightarrow, Q_F)$ be a TBA and

$$w = \sigma_1, \sigma_2, \sigma_3, \ldots$$

a word for $\text{Expr}_B(X)$. An infinite sequence

$$\rho = q_0, q_1, q_2, \ldots \in Q^\omega$$

is called run of $B$ over $w$ under valuation $\beta : X \rightarrow \mathcal{P}(X)$ if and only if

- $q_0 = q_{\text{ini}}$,
- for each $i \in \mathbb{N}_0$ there is a transition
  $$(q_i, \psi_i, q_{i+1}) \in \rightarrow$$
  such that $\sigma_i \models_\beta \psi_i$.

**Example:**

$B_{\text{sym}}$:

$$\Sigma = \{x \mapsto \mathbb{N}\}$$

```
q1
num(x)    even(x)    odd(x)
  |    |    |
  |    |    |
  |    |    |
  |    |    |
  q2
  |    |    |
  |    |    |
  |    |    |
  |    |    |
  q1
```
The Language of a TBA

Definition.
We say TBA $B = (Expr_B(X), X, Q, q_{ini}, \rightarrow, Q_F)$ accepts the word

$$w = (\sigma_i)_{i \in \mathbb{N}_0} \in (Expr_B \rightarrow B)\omega$$

if and only if $B$ has a run

$$\varphi = (q_i)_{i \in \mathbb{N}_0}$$

over $w$ such that fair (or accepting) states are visited infinitely often by $\varphi$, i.e., such that

$$\forall i \in \mathbb{N}_0 \exists j > i : q_j \in Q_F.$$

We call the set $\mathcal{L}(B) \subseteq (Expr_B \rightarrow B)\omega$ of words that are accepted by $B$ the language of $B$. 
**Definition.**
We say TBA $B = (Expr_B(X), X, Q, q_{ini}, \rightarrow, Q_F)$ **accepts** the word

$$w = (\sigma_i)_{i \in \mathbb{N}_0} \in (Expr_B \rightarrow B)^\omega$$

if and only if $B$ **has** a run

$$\rho = (q_i)_{i \in \mathbb{N}_0}$$

over $w$ such that fair (or accepting) states are **visited infinitely often** by $\rho$, i.e., such that

$$\forall i \in \mathbb{N}_0 \exists j > i : q_j \in Q_F.$$ 

We call the set $L(B) \subseteq (Expr_B \rightarrow B)^\omega$ of words that are accepted by $B$ the **language of $B$**.

---

**Example:**

$$B_{sym}: \quad \Sigma = \{x\} \rightarrow \mathbb{N}$$

- $q_1 \quad even(x)$
- $q_2 \quad odd(x)$
- $q_1 \quad even(x)$

- $q_1 \quad odd(x)$
- $q_2 \quad odd(x)$
- $q_1 \quad even(x)$
Language of UML Model
Recall: A UML model $\mathcal{M} = (\mathcal{C}, \mathcal{D}, \mathcal{M}, \mathcal{O})$ and a structure $\mathcal{D}$ denote a set $\llbracket \mathcal{M} \rrbracket$ of (initial and consecutive) computations of the form

$$(\sigma_0, \varepsilon_0) \xrightarrow{a_0} (\sigma_1, \varepsilon_1) \xrightarrow{a_1} (\sigma_2, \varepsilon_2) \xrightarrow{a_2} \ldots$$

where

$$a_i = (\text{cons}_i, \text{Snd}_i, u_i) \in 2^{\mathcal{D}(\mathcal{E})} \times 2^{(\mathcal{D}(\mathcal{E}) \cup \{\ast, +\}) \times \mathcal{D}(\mathcal{C})} \times \mathcal{D}(\mathcal{C}).$$

$$= : \tilde{A}$$
The Language of a Model

Recall: A UML model \( M = (C, D, SM, OD) \) and a structure \( D \) denote a set \([M]\) of (initial and consecutive) computations of the form

\[
(\sigma_0, \varepsilon_0) \xrightarrow{a_0} (\sigma_1, \varepsilon_1) \xrightarrow{a_1} (\sigma_2, \varepsilon_2) \xrightarrow{a_2} \ldots \text{ where}
\]

\[
a_i = (\text{cons}_i, \text{Snd}_i, u_i) \in 2^{D(E)} \times 2^{(D(E) \cup \{*,+\}) \times D(C) \times D(C)} =: \tilde{A}
\]

For the connection between models and interactions, we disregard the configuration of the ether, and define as follows:

** Definition.** Let \( M = (C, D, SM, OD) \) be a UML model and \( D \) a structure. Then

\[
\mathcal{L}(M) := \{(\sigma_i, u_i, \text{cons}_i, \text{Snd}_i)_{i \in \mathbb{N}_0} \in (\Sigma D \times \tilde{A}) \omega \mid

\exists (\varepsilon_i)_{i \in \mathbb{N}_0} : (\sigma_0, \varepsilon_0) \xrightarrow{(\text{cons}_0, \text{Snd}_0)} u_0 \xrightarrow{\sigma_1, \varepsilon_1} \ldots \in [M]\}
\]

is the language of \( M \).
Example: Language of a Model

\[ \mathcal{L}(M) := \{ (\sigma_i, u_i, cons_i, Snd_i)_{i \in \mathbb{N}_0} \mid \exists (\varepsilon_i)_{i \in \mathbb{N}_0} : (\sigma_0, \varepsilon_0) \xrightarrow{(cons_0, Snd_0)} u_0 \xrightarrow{(\sigma_1, \varepsilon_1)} \cdots \in [M] \} \]

**CD:**

\[ \xymatrix{ C_1 \ar[r]^{itsC_2}_{0,1} & C_2 \ar[r]^{itsC_3}_{0,1} & C_3 } \]

**σ₀:**

\[ \xymatrix{ c_1 : C_1 \ar[r]^{itcC_1} & c_2 : C_2 \ar[r]^{itcC_2}_{k = 27} & c_3 : C_3 } \]

\[ (\sigma, \varepsilon) \xrightarrow{(cons, Snd)} \cdots \xrightarrow{u} (\sigma_0, \varepsilon_0) \xrightarrow{(cons_0, Snd_0)} (\sigma_1, \varepsilon_1) \xrightarrow{(cons_1, \{(:E, c_2)\})}_{c_1} (\sigma_2, \varepsilon_2) \xrightarrow{(\{E\}, Snd_2)}_{c_2} \]

\[ (\sigma_3, \varepsilon_3) \xrightarrow{(cons_3, \{(:F, c_3)\})}_{c_2} (\sigma_4, \varepsilon_4) \xrightarrow{(cons_4, \{((G(), c_1)\})}_{c_2} (\sigma_5, \varepsilon_5) \xrightarrow{(\{F\}, Snd_5)}_{c_3} (\sigma_6, \varepsilon_6) \xrightarrow{\cdots} \]
**Words over Signature**

**Definition.** Let $\mathcal{I} = (T, C, V, atr, E)$ be a signature and $\mathcal{D}$ a structure of $\mathcal{I}$. A **word** over $\mathcal{I}$ and $\mathcal{D}$ is an infinite sequence

$$(\sigma_i, u_i, cons_i,_snd_i)_{i \in \mathbb{N}_0} \in \sum^\mathcal{D} \times \mathcal{D}(C) \times 2^{\mathcal{D}(E)} \times 2^{(\mathcal{D}(E) \cup \{\ast, +\}) \times \mathcal{D}(C)}$$

- The language $\mathcal{L}(M)$ of a UML model $M = (CD, SM, OD)$ is a word over the signature $\mathcal{I}(CD)$ induced by $CD$ and $\mathcal{D}$, given structure $\mathcal{D}$. 
Let \((\sigma, u, \text{cons}, \text{Snd}) \in \Sigma \times \tilde{A}\) be a tuple consisting of \textit{system state}, \textit{object identity}, \textit{consume set}, and \textit{send set}.

Let \(\beta : X \rightarrow \mathcal{D}(\mathcal{C})\) be a valuation of the logical variables.

Then

- \((\sigma, u, \text{cons}, \text{Snd}) \models \beta \ \text{true}\)
- \((\sigma, u, \text{cons}, \text{Snd}) \models \beta \ \psi\ \text{if and only if} \ I[\psi](\sigma, \beta) = 1\)
- \((\sigma, u, \text{cons}, \text{Snd}) \models \beta \ \neg \psi\ \text{if and only if} \ \neg (\sigma, \text{cons}, \text{Snd}) \models \beta \ \psi\)
- \((\sigma, u, \text{cons}, \text{Snd}) \models \beta \ \psi_1 \lor \psi_2\ \text{if and only if} \ (\sigma, u, \text{cons}, \text{Snd}) \models \beta \ \psi_1\ \text{or} \ (\sigma, u, \text{cons}, \text{Snd}) \models \beta \ \psi_2\)
- \((\sigma, u, \text{cons}, \text{Snd}) \models \beta \ E^!_{x, y}\ \text{if and only if} \ \beta(x) = u \land \exists e \in \mathcal{D}(E) \bullet (e, \beta(y)) \in \text{Snd}\)
- \((\sigma, u, \text{cons}, \text{Snd}) \models \beta \ E^?_{x, y}\ \text{if and only if} \ \beta(y) = u \land \text{cons} \subset \mathcal{D}(E)\)
Satisfaction of Signal and Attribute Expressions

- Let \((\sigma, u, \text{cons}, \text{Snd}) \in \Sigma_\mathcal{A} \times \tilde{A}\) be a tuple consisting of \text{system state}, \text{object identity}, \text{consume set}, and \text{send set}.
- Let \(\beta : X \rightarrow \mathcal{D}(\mathcal{C})\) be a valuation of the logical variables.

Then

- \((\sigma, u, \text{cons}, \text{Snd}) \models_\beta \text{true}\)
- \((\sigma, u, \text{cons}, \text{Snd}) \models_\beta \psi\) if and only if \(I[\psi](\sigma, \beta) = 1\)
- \((\sigma, u, \text{cons}, \text{Snd}) \models_\beta \neg \psi\) if and only if not \((\sigma, \text{cons}, \text{Snd}) \models_\beta \psi\)
- \((\sigma, u, \text{cons}, \text{Snd}) \models_\beta \psi_1 \lor \psi_2\) if and only if \((\sigma, u, \text{cons}, \text{Snd}) \models_\beta \psi_1\) or \((\sigma, u, \text{cons}, \text{Snd}) \models_\beta \psi_2\)
- \((\sigma, u, \text{cons}, \text{Snd}) \models_\beta E^!_{x,y}\) if and only if \(\beta(x) = u \land \exists e \in \mathcal{D}(E) \bullet (e, \beta(y)) \in \text{Snd}\)
- \((\sigma, u, \text{cons}, \text{Snd}) \models_\beta E^?_{x,y}\) if and only if \(\beta(y) = u \land \text{cons} \subset \mathcal{D}(E)\)

\textbf{Observation:} we don’t use all information from the computation path.
We could, e.g., also keep track of event identities between send and receive.
Example: Model Language and Signal / Attribute Expressions

\[ \begin{align*}
\text{CD:} & \\
C_1 & \xrightarrow{itsC_2} C_2 \xrightarrow{itsC_3} C_3 \\
C_1 & \xrightarrow{C} k : \text{Int} \\
\end{align*} \]

\[ \begin{align*}
\sigma_0 : & \\
\text{itcC}_1 & \xrightarrow{c_1} c_2 : C_2 \xrightarrow{c_2} C_3 \\
k & \xrightarrow{c_3} 27 \\
\end{align*} \]

\[ (\sigma, \varepsilon) \xrightarrow{(\text{cons}, \text{Snd})_u} \cdots \xrightarrow{\text{cons}_{0}, \text{Snd}_0_{u_0}} (\sigma_0, \varepsilon_0) \xrightarrow{\text{cons}_{1}, \{(E, c_2)\}}_c (\sigma_1, \varepsilon_1) \xrightarrow{\{(E, Snd_{0})\}}_c c_2 \\
(\sigma_3, \varepsilon_3) \xrightarrow{\text{cons}_{3}, \{(F, c_3)\}}_{c_2} (\sigma_4, \varepsilon_4) \xrightarrow{\text{cons}_{4}, \{(G(), c_1)\}}_{c_2} (\sigma_5, \varepsilon_5) \xrightarrow{\{(F), Snd_{5}\}}_{c_3} (\sigma_6, \varepsilon_6) \xrightarrow{\cdots} \\
\]

- \( \beta = \{ x \mapsto c_1, y \mapsto c_2, z \mapsto c_3 \} \)
- \( (\sigma_0, u_0, \text{cons}_0, \text{Snd}_0) \models \beta \) \( y.k > 0 \)
- \( (\sigma_0, u_0, \text{cons}_0, \text{Snd}_0) \models \beta \) \( x.k > 0 \)
- \( (\sigma_1, c_1, \text{cons}_1, \{(E, c_2)\}) \models \beta \) \( E^l_{x,y} \)
- \( (\sigma_1, c_1, \text{cons}_1, \{(E, c_2)\}) \models \beta \) \( F^l_{x,y} \)
- \( \cdots \models \beta \) \( E^?_{x,y} \)
- We set \( (\sigma_4, c_2, \text{cons}_4, \{G(), c_1\}) \models \beta \) \( G_{y,x}! \land G_{y,x}? \) (triggered operation or method call).
Definition. A TBA

\[ \mathcal{B} = (\text{Expr}_B(X), X, Q, q_{\text{ini}}, \rightarrow, Q_F) \]

where \( \text{Expr}_B(X) \) is the set of signal and attribute expressions \( \text{Expr}_\mathcal{I}(\mathcal{E}, X) \) over signature \( \mathcal{I} \) is called TBA over \( \mathcal{I} \).
Recall: The TBA $\mathcal{B}(\mathcal{L})$ of LSC $\mathcal{L}$ is $(\text{Expr}_\mathcal{B}(X), X, Q, q_{\text{ini}}, \rightarrow, Q_F)$ with

- $Q$ is the set of cuts of $\mathcal{L}$, $q_{\text{ini}}$ is the instance heads cut,
- $\text{Expr}_\mathcal{B} = \Phi \cup \mathcal{E}_? (X)$,
- $\rightarrow$ consists of loops, progress transitions (from $\rightsquigarrow_F$), and legal exits (cold cond./local inv.),
- $F = \{ C \in Q \mid \Theta(C) = \text{cold} \lor C = L \}$ is the set of cold cuts.

So in the following, we “only” need to construct the transitions’ labels:

$$\rightarrow = \{(q, \psi_{\text{loop}}(q), q) \mid q \in Q\} \cup \{(q, \psi_{\text{prog}}(q, q'), q') \mid q \rightsquigarrow_F q'\} \cup \{(q, \psi_{\text{exit}}(q), L) \mid q \in Q\}$$
\[ \varphi \in \text{OCL} \]

\[ \mathcal{S} = (\mathcal{T}, \mathcal{C}, V, \text{atr}), \text{SM} \]

\[ M = (\sum_{\mathcal{S}}, A_{\mathcal{S}}, \rightarrow_{\text{SM}}) \]

\[ \pi = (\sigma_0, \varepsilon_0) \xrightarrow{(\text{cons}_0, \text{Snd}_0)} u_0 (\sigma_1, \varepsilon_1) \cdots \]

\[ w_\pi = ((\sigma_i, \text{cons}_i, \text{Snd}_i))_{i \in \mathbb{N}} \]

\[ B = (Q_{SD}, q_0, A_{\mathcal{S}}, \rightarrow_{SD}, F_{SD}) \]

\[ G = (N, E, f) \]

\[ \mathcal{O}_D \]
Live Sequence Charts — Semantics Cont’d
**Full LSCs**

A full LSC $\mathcal{L} = (((L, \preceq, \sim), \mathcal{I}, \text{Msg}, \text{Cond}, \text{LocInv}, \Theta), ac_0, am, \Theta_{\mathcal{L}})$ consists of

- **body** ($(L, \preceq, \sim), \mathcal{I}, \text{Msg}, \text{Cond}, \text{LocInv}, \Theta)$,
- **activation condition** $ac_0 \in \text{Expr}_{\mathcal{L}}$,
- **strictness flag** $\text{strict}$ (if false, $\mathcal{L}$ is called **permissive**)
- **activation mode** $am \in \{\text{initial}, \text{invariant}\}$,
- **chart mode** **existential** ($\Theta_{\mathcal{L}} = \text{cold}$) or **universal** ($\Theta_{\mathcal{L}} = \text{hot}$).
A **full LSC** $\mathcal{L} = (((L, \preceq, \sim), \mathcal{I}, \text{Msg}, \text{Cond}, \text{LocInv}, \Theta), ac_0, am, \Theta_\mathcal{L})$ consists of

- **body** $((L, \preceq, \sim), \mathcal{I}, \text{Msg}, \text{Cond}, \text{LocInv}, \Theta)$,
- **activation condition** $ac_0 \in \text{Expr}_{\mathcal{L}}$,
- **strictness flag** $\text{strict}$ (if false, $\mathcal{L}$ is called **permissive**)
- **activation mode** $am \in \{\text{initial, invariant}\}$,
- **chart mode** **existential** ($\Theta_\mathcal{L} = \text{cold}$) or **universal** ($\Theta_\mathcal{L} = \text{hot}$).

**Concrete syntax:**

![Concrete Syntax Diagram]
**Full LSCs**

A full LSC \( \mathcal{L} = (((L, \preceq, \sim), I, \text{Msg}, \text{Cond}, \text{LocInv}, \Theta), ac_0, am, \Theta_{\mathcal{L}}) \) consists of

- **body** \( ((L, \preceq, \sim), I, \text{Msg}, \text{Cond}, \text{LocInv}, \Theta) \),
- **activation condition** \( ac_0 \in \text{Expr}_{\mathcal{L}} \),
- **strictness flag** \( \text{strict} \) (if false, \( \mathcal{L} \) is called **permissive**)
- **activation mode** \( am \in \{\text{initial, invariant}\} \),
- **chart mode** **existential** \( (\Theta_{\mathcal{L}} = \text{cold}) \) or **universal** \( (\Theta_{\mathcal{L}} = \text{hot}) \).

A set of words \( W \subseteq (\text{Expr}_{B} \rightarrow B)^{\omega} \) is **accepted** by \( \mathcal{L} \) if and only if

<table>
<thead>
<tr>
<th>( \Theta_{\mathcal{L}} )</th>
<th>( am = \text{initial} )</th>
<th>( am = \text{invariant} )</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>cold</strong></td>
<td>( \exists w \in W \bullet w^{0} \models ac \land \neg \psi_{\text{exit}}(C_{0}) \land w^{0} \models \psi_{\text{prog}}(\emptyset, C_{0}) \land w/1 \in \mathcal{L}(B(\mathcal{L})) )</td>
<td>( \exists w \in W \exists k \in \mathbb{N}<em>{0} \bullet w^{k} \models ac \land \neg \psi</em>{\text{exit}}(C_{0}) \land w^{k} \models \psi_{\text{prog}}(\emptyset, C_{0}) \land w/k + 1 \in \mathcal{L}(B(\mathcal{L})) )</td>
</tr>
<tr>
<td><strong>hot</strong></td>
<td>( \forall w \in W \bullet w^{0} \models ac \land \neg \psi_{\text{exit}}(C_{0}) \models \psi_{\text{prog}}(\emptyset, C_{0}) \land w/1 \in \mathcal{L}(B(\mathcal{L})) )</td>
<td>( \forall w \in W \forall k \in \mathbb{N}<em>{0} \bullet w^{k} \models ac \land \neg \psi</em>{\text{exit}}(C_{0}) \models \psi_{\text{hot}}^{\text{Cond}}(\emptyset, C_{0}) \land w/k + 1 \in \mathcal{L}(B(\mathcal{L})) )</td>
</tr>
</tbody>
</table>

where \( C_{0} \) is the minimal (or **instance heads**) cut.
Full LSC Semantics: Example

\[
(\sigma, \varepsilon) \xrightarrow{(\text{cons}, \text{Snd})} \cdots \xrightarrow{u} (\sigma_0, \varepsilon_0) \xrightarrow{(\text{cons}_0, \text{Snd}_0)} (\sigma_1, \varepsilon_1) \xrightarrow{(\text{cons}_1, \{(:E, c_2)\})} (\sigma_2, \varepsilon_2) \xrightarrow{c_2} (\sigma_3, \varepsilon_3) \xrightarrow{(\text{cons}_3, \{(:F, c_3)\})} (\sigma_4, \varepsilon_4) \xrightarrow{(\text{cons}_4, \{G() , c_1\})} (\sigma_5, \varepsilon_5) \xrightarrow{c_3} (\sigma_6, \varepsilon_6) \xrightarrow{(\{F\} , \text{Snd}_5)} \cdots 
\]
Note: Activation Condition

\[ L_1 \]
AC: \( c_1 \)
AM: initial \( l: \) permissive

: \( C_1 \)  
: \( C_2 \)  
: \( C_3 \)  

\( E \rightarrow G \)
\( F \rightarrow G \)

\[ L_1 \]
AM: initial \( l: \) permissive

: \( C_1 \)  
: \( C_2 \)  
: \( C_3 \)  

\( E \rightarrow c_1 \)
\( F \rightarrow c_1 \)

\( G \rightarrow c_1 \)
Existential LSC Example: Buy A Softdrink

LSC: buy softdrink
AC: true
AM: invariant I: permissive

User

Vend. Ma.

E1

pSOFT

SOFT
Existential LSC Example: Get Change

LSC: get change
AC: true
AM: invariant I: permissive

User ➔ Vend. Ma.

C50 ➔ E1

pSOFT ➔ SOFT

chg-C50 ➔
Plan:

(i) Given an LSC $\mathcal{L}$ with body

$$((L, \preceq, \sim), \mathcal{I}, \text{Msg}, \text{Cond}, \text{LocInv}, \Theta),$$

(ii) construct a TBA $B_\mathcal{L}$, and

(iii) define language $\mathcal{L}(\mathcal{L})$ of $\mathcal{L}$ in terms of $\mathcal{L}(B_\mathcal{L})$,

in particular taking activation condition and activation mode into account.

(iv) define language $\mathcal{L}(\mathcal{M})$ of a UML model.

• Then $\mathcal{M} \models \mathcal{L}$ (universal) if and only if $\mathcal{L}(\mathcal{M}) \subseteq \mathcal{L}(\mathcal{L})$.

And $\mathcal{M} \models \mathcal{L}$ (existential) if and only if $\mathcal{L}(\mathcal{M}) \cap \mathcal{L}(\mathcal{L}) \neq \emptyset$. 
Live Sequence Charts — Precharts
A full LSC $\mathcal{L} = (PC, MC, ac_0, am, \Theta_\mathcal{L})$ actually consist of

- **pre-chart** $PC = ((L_P, \preceq_P, \sim_P), I_P, \mathcal{I}, Msg_P, Cond_P, LocInv_P, \Theta_P)$ (possibly empty),
- **main-chart** $MC = ((L_M, \preceq_M, \sim_M), I_M, \mathcal{I}, Msg_M, Cond_M, LocInv_M, \Theta_M)$ (non-empty),
- **activation condition** $ac_0 : \text{Bool} \in \text{Expr}_\mathcal{L}$,
- **strictness flag** $\text{strict}$ (otherwise called permissive)
- **activation mode** $am \in \{\text{initial, invariant}\}$,
- **chart mode** existential ($\Theta_\mathcal{L} = \text{cold}$) or universal ($\Theta_\mathcal{L} = \text{hot}$).
### Pre-Charts Semantics

#### LSC: buy water

#### AC: true

#### AM: invariant \( l: \text{strict} \)

<table>
<thead>
<tr>
<th>Role</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>C50</td>
</tr>
<tr>
<td>CoinValidator</td>
<td>pWATER</td>
</tr>
<tr>
<td>( cp ) : ChoicePanel</td>
<td>( cp \rightarrow \text{water_in_stock} )</td>
</tr>
<tr>
<td>Dispenser</td>
<td>dWATER</td>
</tr>
</tbody>
</table>

#### \( am = \text{initial} \)

\[
\begin{align*}
\exists w \in W & \exists m \in N_0 \cdot \\
& \land w^0 \models ac \land \neg \psi_{\text{exit}}(C^P_0) \land \psi_{\text{prog}}(\emptyset, C^P_0) \\
& \land w^1, \ldots, w^m \in L(\mathcal{B}(PC)) \\
& \land w^{m+1} \models \neg \psi_{\text{exit}}(C^M_0) \\
& \land w^{m+1} \models \psi_{\text{prog}}(\emptyset, C^M_0) \\
& \land w/m + 2 \in L(\mathcal{B}(MC))
\end{align*}
\]

#### \( am = \text{invariant} \)

\[
\begin{align*}
\exists w \in W & \exists k < m \in N_0 \cdot \\
& \land w^k \models ac \land \neg \psi_{\text{exit}}(C^P_0) \land \psi_{\text{prog}}(\emptyset, C^P_0) \\
& \land w^{k+1}, \ldots, w^m \in L(\mathcal{B}(PC)) \\
& \land w^{m+1} \models \neg \psi_{\text{exit}}(C^M_0) \\
& \land w^{m+1} \models \psi_{\text{prog}}(\emptyset, C^M_0) \\
& \land w/m + 2 \in L(\mathcal{B}(MC))
\end{align*}
\]

#### \( \Theta_{L} = \text{cold} \)

\[
\begin{align*}
\forall w \in W & \forall m \in N_0 \cdot \\
& \land w^0 \models ac \land \neg \psi_{\text{exit}}(C^P_0) \land \psi_{\text{prog}}(\emptyset, C^P_0) \\
& \land w^1, \ldots, w^m \in L(\mathcal{B}(PC)) \\
& \land w^{m+1} \models \neg \psi_{\text{exit}}(C^M_0) \\
& \implies w^{m+1} \models \psi_{\text{prog}}(\emptyset, C^M_0) \\
& \land w/m + 2 \in L(\mathcal{B}(MC))
\end{align*}
\]

#### \( \Theta_{L} = \text{hot} \)

\[
\begin{align*}
\forall w \in W & \forall k \leq m \in N_0 \cdot \\
& \land w^k \models ac \land \neg \psi_{\text{exit}}(C^P_0) \land \psi_{\text{prog}}(\emptyset, C^P_0) \\
& \land w^{k+1}, \ldots, w^m \in L(\mathcal{B}(PC)) \\
& \land w^{m+1} \models \neg \psi_{\text{exit}}(C^M_0) \\
& \implies w^{m+1} \models \psi_{\text{prog}}(\emptyset, C^M_0) \\
& \land w/m + 2 \in L(\mathcal{B}(MC))
\end{align*}
\]
Universal LSC: Example

LSC: buy water
AC: true
AM: invariant I: strict

User → CoinValidator → cp :ChoicePanel → Dispenser

C50
pWATER

cp -> water_in_stock

dWATER
OK
Universal LSC: Example

LSC: buy water
AC: true
AM: invariant I: strict

User ───> CoinValidator ───> cp : ChoicePanel ───> Dispenser

¬(C50 ∨ E1 ∨ pSOFT! ∨ pTEA! ∨ pFILLUP!)

cp -> water_in_stock

dWATER

OK
Universal LSC: Example

LSC: buy water
AC: true
AM: invariant I: strict

User -> CoinValidator -> cp : ChoicePanel -> Dispenser

C50

pWATER

cp -> water_in_stock

dWATER

OK

¬(dSoft ! ∨ dTEA!)

¬(C50! ∨ E1! ∨ pSOFT! ∨ pTEA! ∨ pFILLUP!)

¬(dWATER)
Forbidden Scenario Example: Don’t Give Two Drinks

[Diagram of a vending machine with the options 'Water', 'Soft', and 'Tea']
Forbidden Scenario Example: Don’t Give Two Drinks

LSC: only one drink
AC: true
AM: invariant I: permissive

User -> Vend. Ma.

E1

pSOFT

SOFT

SOFT

¬C50! ∧ ¬E1!

false
**Existential** LSCs* may hint at test-cases for the acceptance test!

(*: as well as (positive) scenarios in general, like use-cases)
**Note: Sequence Diagrams and (Acceptance) Test**

- **Existential** LSCs* may hint at test-cases for the acceptance test!
  (*: as well as (positive) scenarios in general, like use-cases)

- **Universal** LSCs (and negative/anti-scenarios) in general need exhaustive analysis!
Existential LSCs\(^*\) may hint at **test-cases** for the **acceptance test**!

\(^*\): as well as (positive) scenarios in general, like use-cases

**Universal** LSCs (and negative/anti-scenarios) in general need **exhaustive analysis**!

(Because they require that the software **never ever** exhibits the unwanted behaviour.)
**Plan:**

(i) Given an LSC \( \mathcal{L} \) with body

\[
((L, \preceq, \sim), \mathcal{I}, \text{Msg}, \text{Cond}, \text{LocInv}, \Theta),
\]

(ii) construct a TBA \( B_{\mathcal{L}} \), and

(iii) define language \( \mathcal{L}(\mathcal{L}) \) of \( \mathcal{L} \) in terms of \( \mathcal{L}(B_{\mathcal{L}}) \),

in particular taking activation condition and activation mode into account.

(iv) define language \( \mathcal{L}(\mathcal{M}) \) of a UML model.

- Then \( \mathcal{M} \models \mathcal{L} \) (universal) if and only if \( \mathcal{L}(\mathcal{M}) \subseteq \mathcal{L}(\mathcal{L}) \).
- And \( \mathcal{M} \models \mathcal{L} \) (existential) if and only if \( \mathcal{L}(\mathcal{M}) \cap \mathcal{L}(\mathcal{L}) \neq \emptyset \).
\( \varphi \in \text{OCL} \)

**Model**

\( \mathcal{I} = (\mathcal{F}, \mathcal{C}, V, \text{atr}), \mathcal{M} \)

\( \mathcal{M} = (\sum_{\mathcal{I}}, A_{\mathcal{I}}, \rightarrow_{\mathcal{SM}}) \)

\( \pi = (\sigma_0, \varepsilon_0) \xrightarrow{(\text{cons}_0, \text{Snd}_0)} (\sigma_1, \varepsilon_1) \cdot \cdot \cdot \)

\( w_\pi = ((\sigma_i, \text{cons}_i, \text{Snd}_i))_{i \in \mathbb{N}} \)

**Instances**

\( \mathcal{G} = (N, E, f) \)

\( \mathcal{O}D \)

**Mathematics**

\( CD, SM \)

\( CD, SD \)

\( B = (Q_{SD}, q_0, A_{\mathcal{I}}, \rightarrow_{SD}, F_{SD}) \)

\( \varphi \in \text{OCL} \)

\( \varphi \in \text{OCL} \)

\( \varphi \in \text{OCL} \)
Tell Them What You’ve Told Them...

- **Büchi automata** accept infinite words
  - if there exists is a run over the word,
  - which visits an accepting state infinitely often.

- The language of a model is just a rewriting of computations into words over an alphabet.

- An LSC accepts a word (of a model) if
  - **Existential**: at least one word (of the model) is accepted by the constructed TBA,
  - **Univerison**: all words (of the model) are accepted.

- Activation mode **initial** activates at system startup (only), **invariant** with each satisfied activation condition (or pre-chart).

- **Pre-charts** can be used to state forbidden scenarios.

- **Sequence Diagrams** can be useful for testing.
References
References


