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

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

February, 17th: The Exam.

Semantic Variation Points

Optimistic view:

• tools exist with complete and consistent code generation.

• good modelling-guidelines can contribute to avoiding misunderstandings.

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.

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)
Definition.

\[ LSC \text{ Body} \]

An LSC body over signature \( S = (T, C, V, atr, E) \) is a tuple \((\langle L, \preceq, \sim \rangle, I, \text{Msg}, \text{Cond}, \text{LocInv}, \Theta)\) where

- \( L \) is a finite, non-empty set of locations,
- \( \preceq \subseteq L \times L \) is a partial order,
- \( \sim \subseteq L \times L \) is a symmetric simultaneity relation disjoint with \( \preceq \), i.e. \( \preceq \cap \sim = \emptyset \),
- \( I = \{ I_1, ..., I_n \} \) is a partitioning of \( L \); elements of \( I \) are called instance line,
- \( \text{Msg} \subseteq L \times E \times L \) is a set of messages with \((l, E, l') \in \text{Msg}\) only if \((l, l') \in \preceq \cup \sim\); message \((l, E, l')\) is called instantaneous iff \( l \sim l' \) and asynchronous otherwise,
- \( \text{Cond} \subseteq (2^L \setminus \emptyset) \times \text{Expr}_S \) is a set of conditions with \((L, \phi) \in \text{Cond}\) only if \( l \sim l' \) for all \( l \neq l' \in L \),
- \( \text{LocInv} \subseteq L \times \{ \circ, \cdot \} \times \text{Expr}_S \times L \times \{ \circ, \cdot \} \) is a set of local invariants with \((l, \iota, \phi, l', \iota') \in \text{LocInv}\) only if \( l \prec l' \), \( \circ \) : exclusive, \( \cdot \) : inclusive,
- \( \Theta : L \cup \text{Msg} \cup \text{Cond} \cup \text{LocInv} \rightarrow \{ \text{hot}, \text{cold} \} \) assigns to each location and each element a temperature.
Live Sequence Charts — The Computation
Definition. Let \((L, \preceq, \sim)\), \(I\), \(Msg\), \(Cond\), \(LocInv\), \(\Theta\) 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. \(\forall l, l' \cdot l' \in C \land l \preceq l' \Rightarrow l \in C\),
- it is closed under simultaneity, i.e. \(\forall l, l' \cdot l' \in C \land l \sim l' \Rightarrow l \in C\), and
- it comprises at least one location per instance line, i.e. \(\forall i \in I \exists l \in C \cdot i \cdot l\).

The temperature function is extended to cuts as follows:

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

that is, \(C\) is hot if and only if at least one of its maximal elements is hot.

Cut Examples

- \(C_1\)
- \(C_2\)
- \(C_3\)
- \(\phi\)
- \(E\)
- \(F\)
- \(G\)
A Successor Relation on Cuts

The partial order "⪯" and the simultaneity relation "∼" of locations induce a direct successor relation on cuts of an LSC body as follows:

**Definition.** Let $C \subseteq L$ be a cut of LSC body $(L, \preceq, \sim, I, \text{Msg}, \text{Cond}, \text{LocInv}, \Theta)$. A set $\emptyset \neq F \subseteq L$ of locations is called a fired-set $F$ of cut $C$ if and only if

1. $C \cap F = \emptyset$ and $C \cup F$ is a cut, i.e., $F$ is closed under simultaneity,
2. all locations in $F$ are direct $\prec$-successors of the front of $C$, i.e., $\forall l \in F \exists l' \in C : l' \prec l \land (\nexists l'' \in C : l' \prec l'' \prec l)$,
3. locations in $F$, that lie on the same instance line, are pairwise unordered, i.e., $\forall l \neq l' \in F : (\exists I \in I : \{l, l'\} \subseteq I) \Rightarrow l \not\preceq l'$ and $l' \not\preceq l$,
4. for each asynchronous (!) message reception in $F$, the corresponding sending is already in $C$, i.e., $\forall (l, E, l') \in \text{Msg} : l' \in F \Rightarrow l \in C$.

The cut $C' = C \cup F$ is called the direct successor of $C$ via $F$, denoted by $C \xrightarrow{F} C'$.
• Let $S = (T, C, V, atr, E)$ be a signature and $X$ a set of logical variables,

• The signal and attribute expressions $Expr_S(E, X)$ are defined by the grammar:

\[
\psi ::= \text{true} | \psi | E!x,y | E?x,y | \neg \psi | \psi_1 \lor \psi_2,
\]

where $\text{expr} : \text{Bool} \in Expr_S$, $E \in E$, $x,y \in X$ (or keyword `env`).

• We use $E!?X := \{ E!x,y, E?x,y \mid E \in E, x,y \in X \}$ to denote the set of event expressions over $E$ 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,
• $Expr_B = \Phi \dot{\cup} E!?X$,
• $\rightarrow$ consists of loops, progress transitions (from $\xRightarrow{} F$), 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.
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

**Definition.**

A Symbolic Büchi Automaton (TBA) is a tuple $B = (\mathcal{X}, \mathcal{E}_{\mathbb{B}}(\mathcal{X}), \mathcal{Q}, \mathcal{q}_{\text{ini}}, \rightarrow, \mathcal{Q}_F)$ where

- $\mathcal{X}$ is a set of logical variables,
- $\mathcal{E}_{\mathbb{B}}(\mathcal{X})$ is a set of Boolean expressions over $\mathcal{X}$,
- $\mathcal{Q}$ is a finite set of states,
- $\mathcal{q}_{\text{ini}} \in \mathcal{Q}$ is the initial state,
- $\rightarrow \subseteq \mathcal{Q} \times \mathcal{E}_{\mathbb{B}}(\mathcal{X}) \times \mathcal{Q}$ is the transition relation. Transitions $(q, \psi, q')$ from $q$ to $q'$ are labelled with an expression $\psi \in \mathcal{E}_{\mathbb{B}}(\mathcal{X})$.
- $\mathcal{Q}_F \subseteq \mathcal{Q}$ is the set of fair (or accepting) states.
The Language of a TBA

Let $B$ be the set of logical variables, $\Sigma$ be the alphabet, $X$ be the set of expressions over $\Sigma$, and $\omega$ be a set of infinite sequences of $\Sigma$. We call the set $\omega$ the language of $B$. An infinite sequence is called a word for $B$ if and only if $\omega$ satisfies $D_{\omega}$. Every infinite sequence is a word for $B$ if and only if $\omega$ satisfies $D_{\omega}$. A set of logical variables is called a set of Boolean variables.

Run of TBA

Let $S$ be a TBA and $W$ be a word over $\Sigma$. We call the set $W$ a run of $S$ if and only if $\omega$ satisfies $D_{\omega}$. Every word over $\Sigma$ is a run of $S$ if and only if $\omega$ satisfies $D_{\omega}$. A set of logical variables is called a set of Boolean variables.

The Language of TBA

Let $S$ be a TBA and $W$ be a word over $\Sigma$. We call the set $W$ the language of $S$ if and only if $\omega$ satisfies $D_{\omega}$. Every word over $\Sigma$ is the language of $S$ if and only if $\omega$ satisfies $D_{\omega}$. A set of logical variables is called a set of Boolean variables.
The Language of a Model

Example: Language of a Model

The Language of a Model

Satisfaction of Signal and Structure Equations

Definition.

The Language of a Model
...
Existential LSC Example: Get Change

Existential LSC Example: Buy A Softdrink
Forbidding Scenario Example: Don't Give Two Drinks

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

User

End. Ma.

E
pSOFT

¬
C50
∧ ¬
E1
false

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.)
Then testing can be useful for Sequence Diagrams in particular taking activation condition and activation mode into account.

References


CSDUML 2003