eLTS

Pascal Sotin

14 mai 2009

0.1 Definitions

For a given service s, we assume a set of services sub_s.

eLTS. An eLTS is a 8-uple[1] $\langle Q, G, L, \delta, q_0, Q_F, \Phi, \Psi \rangle$.

- Q is a finite set of states[2].
- L is a set of basic labels containing at least ε.
- G is a set of guards containing at least `true` and `false`.
- $\delta : Q \times G \times L \nrightarrow Q$ is the transition function by guarded basic label (deterministic in state and label).
- $\Psi : Q \times G \times sub_s \nrightarrow Q$ is the transition function by guarded mandatory subservice calls.
- $q_0 \in Q$ is the unique initial state. $Q_F \subseteq Q$ are the final states.
- $\Phi : Q \leftrightarrow sub_s$ is a relation indicating the optional subservice calls.

Mandatory and optional subservice calls have to be done by the caller.

Graphic view. On the graphic view of an eLTS, the mandatory calls Ψ are depicted by transitions labelled by `[[`ss`]]` where $ss \in sub_s$. The optional calls Φ are depicted by labels on the states, of the form `<<`$ss_1, \ldots, ss_2$`>>` where $ss_i \in sub_s$. The guards are depicted by `[`g`]` in front of the transition, where $g \in G$.

LTS. A LTS is a 6-uple $\langle Q, G, L, \delta, q_0, Q_F \rangle$. The description of the elements of the tuple is the same as in an eLTS. The graphical representation is similar to the one of an eLTS (but simpler).

Primitive transition system. A primitive transition system is a 5-uple $\langle \Sigma, L, \rightarrow, I, F \rangle$. Σ is a (potentially infinite) set of states, but unlike the ones of a LTS or eLTS, it can be infinite. L is still a set of label, with $\rightarrow$ a labelled transition system. $I \subseteq \Sigma$ and $F \subseteq \Sigma$ are respectively the set of initial states and final states.

[1] Originally, in SC07, the eLTS is first described by a 6-uple, without Ψ which was introduced latter on in the paper. This choice was made to preserve existing techniques and tools. For simpler model manipulation, this paper includes Ψ in the tuple, exclude `[[`ss`]]` transitions from δ and takes guards out of the labels.

[2] Originally, in SC07, the set Q of states was called S. This paper changes this convention to avoid the confusion state/service.

0.2 Dynamic semantics

In this section we provide the dynamic semantics of primitive TS's, LTS's and eLTS's. The semantics of a primitive TS is a trace semantics. The semantics of a LTS is a primitive TS. The semantics of a eLTS is a LTS.

Primitive TS. The **trace semantics** of a primitive transition system $P = \langle \Sigma, L, \rightarrow, I, F \rangle$ is the set of partial traces $[\![P]\!]_{\mathrm{tr}}$.

$$[\![P]\!]_{\mathrm{tr}} = \{[l_{0.1}, l_{1.2}, \ldots, l_{n-1.n}] \,|\, \sigma_0 \in I \wedge \sigma_i \stackrel{l_{i.i+1}}{\rightarrow} \sigma_{i+1}\}$$

LTS. The **operational semantics** of a LTS $\langle Q, G, L, \delta, q_0, Q_F \rangle$ is described by the primitive transition system $\langle \Sigma \times Q, L, \rightarrow, \Sigma_I \times \{q_0\}, \Sigma \times Q_F \rangle$ where Σ is a set of possible environments[3] (given by the variables of the component) among which we distinguish Σ_I, the set of possible initial environments (given by the initializations).

$$\frac{(q_1, \texttt{[}g\texttt{]}\, l, q_2) \in \delta \qquad [\![g]\!]_{\mathrm{cond}}(\sigma) \qquad \sigma' \in [\![l]\!]_{\mathrm{exec}}(\sigma)}{(\sigma, q_1) \stackrel{l}{\rightarrow} (\sigma', q_2)}$$

With $[\![]\!]_{\mathrm{cond}} : G \rightarrow \Sigma \rightarrow \mathbb{B}$ giving the semantics of the guard in a given environment and $[\![]\!]_{\mathrm{exec}} : L \rightarrow \Sigma \rightarrow \mathcal{P}(\Sigma)$ giving the non-deterministic semantics of the action of the basic labels on the environment.

We name *prim* the function that goes from an LTS to its primitive transition system.

Property 1 (Soundness)**.** *If a service has a LTS Δ, for all monitoring of a call to this service, their exists a trace $t \in [\![prim(\Delta)]\!]_{tr}$ that matches the monitored behavior.*

Property 2 (Completeness)**.** *If a service has a LTS Δ, for all trace $t \in [\![prim(\Delta)]\!]_{tr}$, their exists an assembly containing a component containing this service, such that at some point the monitoring of a call to this service matches the trace t.*

Property 1, that we call soundness, states that whatever the context, no behavior outside of the semantics can occur. Property 2, that we call completeness, states that all the behaviors of the semantics can be observed, provided that we can choose the context. We are talking here of abstract behaviors, and not of concrete behaviors provided by an implementation of the abstract service seen as a specification. The semantics discuss the behavior of the service only, but we have the guarantee that in any context, if we focus on one invocation of the service, no more (and no less) than the semantics can be obtained.

[3] We use the symbol Σ both for a state in a primitive transition system and for an environment in a LTS operational semantics

eLTS. The **operational semantics** of an eLTS of a service $\underline{s}$ within a component containing the set of services $\mathcal{S}$ is defined by a LTS $\langle Q_u, G_u, L_u, \delta_u, q_0, Q_F\rangle$[4]. For a given service s, the eLTS is $\langle Q_s, G_s, L_s, \delta_s, q_0(s), Q_F(s), \Phi_s, \Psi_s\rangle$.

We write $q_1 \overset{\texttt{[}g\texttt{]}\,l}{\hookrightarrow} q_2$ for $(q_1, g, l, q_2) \in \delta_u$ and define this transition system by the following four rules:

$$\text{basic transition}\ \frac{(q_1, g, l, q_2) \in \delta_s}{c.(s : q_1) \overset{\texttt{[}g\texttt{]}\,l}{\hookrightarrow} c.(s : q_2)}$$

$$\text{mandatory call}\ \frac{(q_1, g, ss, q_2) \in \Psi_s}{c.(s : q_1) \overset{\texttt{[}g\texttt{]??}ss}{\hookrightarrow} c.(s : q_2).(ss : q_0(s))}$$

$$\text{optional call}\ \frac{(q_1, ss) \in \Phi_s}{c.(s : q_1) \overset{\texttt{[true]??}ss}{\hookrightarrow} c.(s : q_1).(ss : q_0(s))}$$

$$\text{(sub)service end}\ \frac{q \in Q_F(s)}{c.(s : q) \overset{\texttt{[true]}\varepsilon}{\hookrightarrow} c}$$

The initial state q_0 of the LTS is $\tau.(\underline{s} : q_0(\underline{s}))$. The set of states, guards and labels (Q_u, G_u, L_u) are those collected from q_0 by δ_u. The set of final states Q_F is $\{\tau\}$.

We note that any $(q, ss) \in \Phi$ can be replaced by $(q, \texttt{true}, ss, q) \in \Psi$.

[4]Subscript u stand for unfolded.