Refinement for Transition Systems with Responses[*]

Marco Carbone Thomas Hildebrandt Hugo A. López Gian Perrone Andrzej Wąsowski
IT University of Copenhagen, Denmark
{maca,hilde,lopez,gdpe,wasowski}@itu.dk

1 Introduction

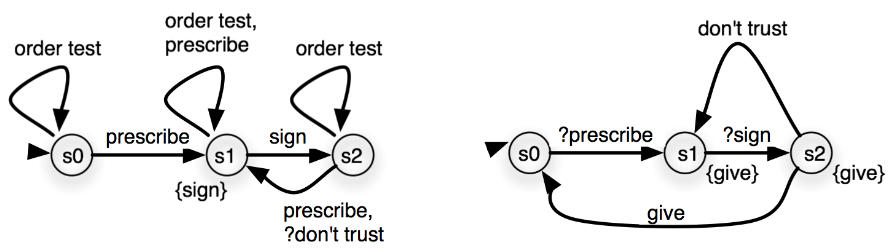
Modal transition systems (MTS) were introduced originally in the seminal work of Larsen and Thomsen [7] (see also [2]) as a basic transition system model supporting stepwise specification and refinement of parallel processes. A MTS can be regarded as a labeled transition system (LTS) in which a subset of the transitions are identified as being required (must), while the others are allowed (may). In a MTS every required transition is also allowed, to avoid inconsistencies. A MTS describes simultaneously an over-approximation and an under-approximation of a process in an intertwined manner. In a stepwise refinement scenario this approximation interval is narrowed down to a single process, an LTS.

Subsequent work has lifted the assumption that required transitions need also be allowed, leading to the model of mixed transition systems [4]. However, the notion of a must transition that is not also a may transition appears quite intricate; it calls for interpreting the specifications at the targets of the must transitions which must *all* be satisfied in conjunction with some choice of may transition. We propose to take a step back and sketch a generalization of MTSs with a restricted kind of must transitions that allow for a simpler semantics. Due to lack of space, we simplify the exposition by restricting our attention to *action-deterministic* transition systems, where for each action a there is at most one a-transition from each state. We propose to replace the must transitions by a set of must *actions* assigned to every state. For readers familiar with mixed transition systems, this resembles a must transition to a "top" state from which every action is possible as a may transition. We refer to this set of actions as the response (or must) set, and we name the resulting model *Transition Systems with Responses* (TSR). We believe the mixed transition systems represented by TSRs are much simpler to understand and work with, and yet they still capture a rich set of specifications. Indeed, TRSs arise as the natural transition system underlying Dynamic Condition Response (DCR) Graphs (e.g. [5, 6]), which generalize event structures to allow finite, executable specifications of ω-regular languages and are particularly useful for specification of flexible workflows where many actions are optional and liveness properties are needed.

Consider the example of Fig. 1, illustrating two parts of a medical workflow described as TSRs. The TSR given in Fig. 1(a) shows that the doctor may first order any number of tests and then prescribe some medicine. Having prescribed medicine, it becomes a requirement to sign the prescription, so the response set of state $s1$ now contains the action sign. The TSR for a nurse given in Fig. 1(b) may be interpreted similarly: If the nurse receives a prescription then the TSR moves to state $s1$ in which give is included in the response set, meaning that the medication must be given. However, this requirement cannot be satisfied in the present state, since there is no outgoing transition labelled with give. This reflects the rule in the workflow that the nurse is not allowed to give medicine before the prescription is signed. If a signature is received, then the nurse still has the requirement to do a give transition, and so can finally

[*]Supported by Danish Council for Strategic Research grants 2106-080046 The Jingling Genies, 2106-07-0019 Trustworthy Pervasive Healthcare Services, and by Villum Kann Rasmussen Foundation through VKR Centre of Excellence MT-LAB.

Submitted to:
Foundations of Interface Technologies 2012



(a) Medication, doctor (b) Medication, nurse

Figure 1: Medication workflow as two interacting transition systems with responses

perform it and return to the initial state. However, the nurse can also choose to do a `don't trust` action, which signals to the doctor that signing must occur again. In the doctor's TSR the `?don't trust` action takes control back to the state where *sign* is required as response.

2 Transition Systems with Responses and Refinement

Definition 1. *A* Mixed Transition System *is a tuple* $T = \langle S, s_0, \mathsf{Act}, \rightarrow_\square, \rightarrow_\diamond \rangle$ *where S is a set of* states, $s_0 \in S$ *is the* initial state, Act *is a set of actions, and* $\rightarrow_\square, \rightarrow_\diamond \subseteq S \times \mathsf{Act} \times S$ *are respectively* must *and* may *transition relations. T is also a* Modal Transition System *(MTS) if additionally* $\rightarrow_\square \subseteq \rightarrow_\diamond$.

Definition 2. *A* Transition System with Responses *(TSR) is a tuple* $T = \langle S, s_0, \mathsf{Act}, \square, \rightarrow \rangle$ *where S,* s_0, Act *are like above and* $\rightarrow \subseteq S \times \mathsf{Act} \times S$ *is a transition relation,* $\square : S \rightarrow \mathscr{P}(\mathsf{Act})$ *defines for each state the* response actions *that must be executed. Let* $\diamond(s) =_{def} \{a \mid \exists s'.s \xrightarrow{a} s'\}$*, i.e., the actions on transitions that may be taken from s. We refer to a finite or infinite sequence of transitions starting at the inital state as a run. A run is accepting if for any intermediate state s in the run,* $a \in \square(s)$ *implies eventually after that state there will be a transition in the run labelled with the action a or a state* s' *where* $a \notin \square(s')$.

Proposition 1. *Action-deterministic modal transition systems correspond to the subset of TSRs where for all states s it holds that* $\square(s) \subseteq \diamond(s)$*. TSRs correspond to the subset of mixed transition systems of the form* $\langle S \uplus \{t\}, s_0 \in S, \mathsf{Act}, \rightarrow_\square \subseteq S \times \mathsf{Act} \times \{t\}, \rightarrow_\diamond = \rightarrow \cup \{t\} \times \mathsf{Act} \times \{t\} \rangle$*, where* $\rightarrow \subseteq S \times \mathsf{Act} \times S$.

An action-deterministic MTS $M = \langle S, s_0, \mathsf{Act}, \rightarrow_\square, \rightarrow_\diamond \rangle$ can be represented as the TSR $R(M) = \langle S, s_0, \mathsf{Act}, \square, \rightarrow_\diamond \rangle$ where $\square(s) =_{def} \{a \mid \exists s' \in S.s \xrightarrow{a} s'\}$ and a TSR $T = \langle S, s_0, \mathsf{Act}, \square, \rightarrow \rangle$ as the action-deterministic MTS $M(T) = \langle S, s_0, \mathsf{Act}, \rightarrow_\square, \rightarrow_\diamond \rangle$, where $\rightarrow_\square = \{(s, a, s') \mid a \in \square(s) \wedge s \xrightarrow{a} s'\}$ and $\rightarrow_\diamond = \rightarrow$. A TSR $T = \langle S, s_0, \mathsf{Act}, \square, \rightarrow \rangle$ corresponds to a mixed transition system $Mix(T) = \langle S \uplus \{t\}, s_0, \mathsf{Act}, \rightarrow_\square, \rightarrow_\diamond \rangle$ where $\rightarrow_\square = \{(s, a, t) \mid a \in \square(s)\}$ and $\rightarrow_\diamond = \rightarrow \cup \{t\} \times \mathsf{Act} \times \{t\}$.

Definition 3 (Deadlock and Liveness)**.** *A* deadlock *state in a TSR* $T = \langle S, s_0, \mathsf{Act}, \square, \rightarrow \rangle$ *is a state with a non-empty* must *set, and no outgoing transitions, i.e., a state in which some actions are required but no further transitions are possible. Formally we define a predicate deadlock on S by* $deadlock(s) \equiv \square(s) \neq \emptyset \wedge \diamond(s) = \emptyset$*. A TSR is* deadlock free *if it has no reachable deadlock state. A* live *state is one from which there exists an accepting run. A TSR is live if all reachable states are live.*

Definition 4 (Refinement)**.** *A binary relation* $\mathscr{R} \subseteq S_1 \times S_2$ *between the state sets of two transition systems with responses* $T_j = \langle S_j, i_j, \mathsf{Act}, \square_j, \rightarrow_j \rangle$ *for* $j \in \{1, 2\}$ *is a refinement if* $i_1 \mathscr{R} i_2$ *and* $s_1 \mathscr{R} s_2$ *implies*

1. $\forall s_1 \xrightarrow{a}_1 s_1'$ *and* $a \in \square_1(s_1)$ *implies* $\exists s_2 \xrightarrow{a}_2 s_2'$, $a \in \square_2(s_2)$ *and* $s_1' \mathscr{R} s_2'$,
2. $\forall s_2 \xrightarrow{a}_2 s_2'$ *implies* $\exists s_1 \xrightarrow{a}_1 s_1'$ *and* $s_1' \mathscr{R} s_2'$

The refinement $\mathscr{R}$ is safe if it reflects deadlock states, so deadlock$(s_2) \implies$ deadlock(s_1) whenever $s_1 \mathscr{R} s_2$.

Proposition 2. *Given two TSRs $T_i = \langle S_i, s_{i,0}, \mathsf{Act}, \Box_i, \rightarrow_i \rangle$ for $i \in \{1,2\}$, if $\mathscr{R} \subseteq S_1 \times S_2$ is a safe refinement then T_2 is deadlock free if T_1 is deadlock free.*

An example of a safe refinement of the TSR in Fig. 1(a) is the TSR obtained by removing order test transitions. Since the first prescribe is not in $\Box$, another example is a TSR with just the initial state.

An example of a non-safe refinement of the TSR in Fig. 1(b) is the TSR obtained by removing transition labelled with ?sign. However, note that this action belongs to the *interface* of the TRS, i.e., it is controlled by the environment. This suggests as a next step studying a variant of refinement for TSRs where the interface actions are preserved akin to partial bisimulation [8] or alternating simulation [1].

3 Conclusion and Future Work

We have introduced Transition Systems with Responses (TSRs) as a new generalization of Modal Transition Systems which represents a restricted class of mixed transition systems that are much simpler than general mixed transition systems, and yet which remain expressive and allow natural definitions of deadlock freedom and liveness for infinite computations. We have proposed a notion of refinement, exemplified by a medical workflow consisting of two interacting TSRs.

We leave as future work the study of deadlock and liveness properties, as well as the detail of refinement and bisimulation for TSRs, and the relation to other models with liveness, such as DCR Graphs in [5, 6] and Harel's Live Sequence Charts (LSCs) [3]. This will include lifting the restriction to action-deterministic systems, which can be done by considering TSRs with transitions carrying labelled events (as in asynchronous transition systems and labelled event structures) and response sets being sets of events, not actions.

References

[1] Luca de Alfaro & Thomas A. Henzinger (2001): *Interface Automata*. In: *Proceedings of the Ninth Annual Symposium on Foundations of Software Engineering (FSE)*, ACM Press, Vienna, Austria, pp. 109–120.

[2] Adam Antonik, Michael Huth, Kim G. Larsen, Ulrik Nyman & Andrzej Wąsowski (2008): *20 Years of Modal and Mixed Specifications*. *Bulletin of EATCS* 95. Available at http://processalgebra.blogspot.com/2008/05/concurrency-column-for-beatcs-june-2008.html.

[3] W. Damm & D. Harel (2001): *LSCs: Breathing Life into Message Sequence Charts*. *Formal Methods in System Design* 19(1), pp. 45–80.

[4] Dennis Dams (1996): *Abstract Interpretation and Partition Refinement for Model Checking*. Ph.D. thesis, Eindhoven University of Technology.

[5] Thomas Hildebrandt & Raghava Rao Mukkamala (2011): *Declarative Event-Based Workflow as Distributed Dynamic Condition Response Graphs*. In: *Post proceedings of International Workshop on Programming Language Approaches to Concurrency and Communication-cEntric Software (PLACES 10)*. Available at http://www.itu.dk/people/rao/rao_files/dcrsplacescamredver.pdf.

[6] Thomas T. Hildebrandt, Raghava Rao Mukkamala & Tijs Slaats (2011): *Safe Distribution of Declarative Processes*. In: *SEFM*, pp. 237–252. Available at http://dx.doi.org/10.1007/978-3-642-24690-6_17.

[7] Kim Guldstrand Larsen & Bent Thomsen (1988): *A Modal Process Logic*. In: *LICS*, IEEE Computer Society, IEEE Computer Society, pp. 203–210.

[8] J. Rutten (2000): *Coalgebra, concurrency, and control*. *Discrete event systems: analysis and control* 569, p. 31.