ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY
Volume 29, Number 1, 2026, 65–76
doi: 10.59277/ROMJIST.2026.1.06

Behavioural Equivalences in Timed Reactive Systems

Bogdan AMAN[1,2,*] and Gabriel CIOBANU[1]

[1]Institute of Computer Science, Romanian Academy, Iasi Branch, Romania
[2]Faculty of Computer Science, Alexandru Ioan Cuza University, Iasi, Romania
Email: bogdan.aman@iit.academiaromana-is.ro*,
gabriel.ciobanu@iit.academiaromana-is.ro
* Corresponding author

Abstract. This paper introduces a timed process calculus designed to model reactive systems that can dynamically adjust their spatial organization and communication structure. Authors' calculus, named rTIMO, facilitates timed agent migration between distributed locations, with explicit timers governing both communication and movement actions. New behavioural equivalence relations are proposed aiming at identifying behaviours that closely align with desired specifications. To illustrate this, the paper includes an example of resource-sensitive routing, in which a driver aims to reach a specific location. The calculus proposed in this paper provides a formal foundation for compositional reasoning about timed, mobile multi-agent systems.

Key-words: Behavioural equivalence; distributed real-time systems; process calculi.

1. Introduction

Reactive systems are characterized by continuous interaction with their environment, exhibiting dynamic spatial and communication properties. To formalize such systems, process calculi offer a powerful framework emphasizing compositional operators and behavioural equivalences. Building on foundational works such as the π-calculus and timed variants, this paper focuses on integrating explicit timers governing communication and migration with a calculus (rTIMO) that captures reconfiguration through channel and location exchange.

The π-calculus [1] represents the first approach to formal model concurrent systems using process calculi, and using exchanged values on shared channels to adjust the communication structure. Process calculi offer these main advantages: (i) system specifications rely on a few

primitives and operators; (ii) agents communicate names using channels; and (iii) behavioural equivalence together with equational reasoning provide the means to analyze agents. The capacity of process calculi to construct large systems by composing smaller ones in parallel distinguishes them from existing computational models dynamic modal operators [2], dynamic graph games [3], reactive Kripke models [4], reactive systems [5] and their fuzzy version [6].

Timers allow the modelling of systems whose behaviour may evolve not only through explicit actions but also as a consequence of inactivity, as in timed automata [7], $tD\pi$ [8], and TIMO [9]. Note that, these approaches do not provide a natural integration of timed communication and timed mobility in distributed systems. To address this limitation, an extension of TIMO called rTIMO (real-Time Mobility) is proposed in [10] and applied in [11], in which timers explicitly govern communication and migration actions of agents in a distributed setting. It is important to note, though, that rTIMO is not the only formalism exploring the interplay of time and mobility. For example, kiltera language [12] introduces a different treatment of time, where the passage of time results in processes being delayed, or equivalently, scheduled for future execution, in contrast to the explicit use of timers as deadlines in rTIMO. For reactive systems, rTIMO offers the following main advantages: (i) the capability to model timed agent mobility across distributed locations; (ii) support for parallel execution of agents, with communication restricted to co-located agents; and (iii) the application of behavioural equivalence to identify system equivalences.

Authors' current approach compares the behaviours of reactive systems by accounting for the actions performed and their corresponding timers. The selection of a suitable behavioural equivalence relation depends on the reactive system being analyzed and the properties required to be preserved. Since reactive systems may contain parallel subsystems, the proposed equivalence relations should be compositional. To illustrate the functioning of rTIMO, an example from [4] is adapted, modeled in rTIMO, and bisimulations are defined to verify whether the resulting rTIMO models are equivalent.

The paper is structured as follows. Section 2 introduces the syntax of rTIMO and its operational semantics, and provides an example of a reactive system with a driver aiming to reach a target location. Section 3 introduces several bisimulations, analyses their interrelations, and illustrates how they apply to specific reactive systems. The paper ends with details about other approaches derived from TIMO and also provides future research directions.

2. Timed Reactive Systems

The rTIMO calculus enables reactive communication and mobility through the exchange of channels and locations. Timers allow the modelling of systems whose behaviour may evolve not only through explicit actions but also as a consequence of inactivity. In these reactive systems, local communication and agent migration between locations occur in parallel, and a global clock models the passage of time whenever no communication or migration is possible.

Table 1 presents the syntax of rTIMO, under the following premises: (i) there are three sets containing locations l (set Loc), communication channels a (set $Chan$), and agent identifiers id (set Id); (ii) every agent identifier $id \in Id$ is associated with a unique definition $id(u_1, \ldots, u_{m_{id}}) \stackrel{def}{=} P_{id}$; (iii) t, v and u are positive real-timeouts, expressions and variables, respectively.

A migration action is associated with a real-time timer $^{\Delta t}$ specifying the time t that must elapse before an agent can move to a new location and resume execution. For example, an agent $go^{\Delta 5} l$ then P waits for 5 time units at its current location before migrating to location l and

Table 1. rTIMO syntax

Agents	P, Q	::=	$a^{\Delta t}!\langle v\rangle$ then P else Q	(output)
			$a^{\Delta t}?(u)$ then P else Q	(input)
			$go^{\Delta t} l$ then P	(move)
			if *test* then P else Q	(branch)
			$\mathbf{0}$	(termination)
			$id(v)$	(recursion)
			$P \mid Q$	(parallel)
Reactive Systems	N, N'	::=	$l[[P]] \ , \ N \mid N'$	

continuing as P. A communication action is also associated with a real-time timer $^{\Delta t}$, allowing agents to communicate if they are co-located simultaneously and they can send and receive information on a shared channel before their timers expire. For example, an agent $a^{\Delta 3}!\langle v\rangle$ then P else Q can wait up to 3 time units to send v on channel a, and an agent $a^{\Delta 2}?(u)$ then P else Q can wait up to 2 time units on channel a to receive a value in order to replace in P the variable u. The previous two agents proceed with P at the current location if the communication succeeds; otherwise, if a timeout occurs, they proceed with Q at the same location. An agent if $test$ then P else Q evaluates $test$ and if is $true$ it proceeds as P; otherwise it proceeds as Q.

In an agent $a^{\Delta t}?(u)$ then P else Q, the variable u is bound in all parts except in Q. An agent definition $id(u_1, \ldots, u_{m_{id}}) \stackrel{def}{=} P_{id}$ is well-defined only if the set $\{u_1, \ldots, u_{m_{id}}\}$ contains its free variables. An agent $\{v/u\}P$ is derived from P by using value v to instantiate the free variable u. To prevent name conflicts, alpha-conversions of variables which are not free may be necessary.

A flexible migration strategy can be applied, where the system is spatially reorganized through go actions that use location variables instantiated with location names received over communication channels. For example, the agent $a^{\Delta 4}?(l)$ then $go^{\Delta t} l$ then P else Q is able migrate to location l only after receiving a value on channel a to instantiate l. In this way, reactive agents can be defined without predefined migration routes, enabling them to reconfigure their movement dynamically through values communicated between agents. The agent $\mathbf{0}$ completed its execution, the agent $id(v)$ uses the unique definition of id to model recursive behaviour, agents are composed in parallel using the operator $|$, while a reactive system N is composed of located agents, where $l[[P]]$ denotes a location l containing an agent P.

A structural equivalence relation $\equiv$ is introduced as the least congruence satisfying the equalities in Table 2, enabling the rearrangement of agents in a reactive system so that the operational semantics in Table 3 can be applied.

Table 2. rTIMO Structural congruence

$N \mid l[[\mathbf{0}]] \equiv N$	(identity)
$N \mid N' \equiv N' \mid N$	(commutativity)
$(N \mid N') \mid N'' \equiv N \mid (N' \mid N'')$	(associativity)
$l[[P]] \mid l[[Q]] \equiv l[[P \mid Q]]$	(composition)

Table 3 presents the operational semantics of rTIMO, expressed using two kinds of labelled transitions: $N \stackrel{\lambda}{\rightarrow} N'$ and $N \stackrel{t}{\rightsquigarrow} N'$. A labelled transition $N \stackrel{\lambda}{\rightarrow} N'$ indicates that performing the action λ to the reactive system N yields the reactive system N'. The action λ can be $l \lhd l'$ (indicating a movement from location l to location l'), $\{v/u\}@l$ (indicating a communication at

location l of value v to instantiate variable u), $\Delta 0@l$ (indicating an expired timer at location l), $call@l$ (indicating an unfolding at location l, $true@l$ and $false@l$ (indicating that a test performed at location l). A labelled transition $N \overset{t}{\rightsquigarrow} N'$ indicates that after t time units elapse in the reactive system N, the system evolves into N'.

Table 3. rTiMo operational semantics

Name	Condition	Rule
(DStop)	$t > 0$	$l[[\mathbf{0}]] \overset{\mathbf{t}}{\rightsquigarrow} \mathbf{1}[[\mathbf{0}]]$
(Move0)	$t = 0$	$l[[go^{\Delta t} l' \text{ then } P]] \xrightarrow{l \triangleright l'} l'[[P]]$
(DMove)	$t \geq t'$	$l[[go^{\Delta t} l' \text{ then } P]] \overset{t'}{\rightsquigarrow} l[[go^{\Delta t - t'} l' \text{ then } P]]$
(Com)	$t, t' > 0$	$l[[a^{\Delta t}!\langle v\rangle \text{ then } P \text{ else } Q \mid a^{\Delta t'}?(u) \text{ then } P' \text{ else } Q']] \xrightarrow{\{v/u\}@l} l[[P \mid \{v/u\}P']]$
(DPut)	$t \geq t' > 0$	$l[[a^{\Delta t}!\langle v\rangle \text{ then } P \text{ else } Q]] \overset{t'}{\rightsquigarrow} l[[a^{\Delta t - t'}!\langle v\rangle \text{ then } P \text{ else } Q]]$
(Put0)	$t = 0$	$l[[a^{\Delta t}!\langle v\rangle \text{ then } P \text{ else } Q]] \xrightarrow{\Delta 0@l} l[[Q]]$
(DGet)	$t \geq t' > 0$	$l[[a^{\Delta t}?(u) \text{ then } P \text{ else } Q]] \overset{t'}{\rightsquigarrow} l[[a^{\Delta t - t'}?(u) \text{ then } P \text{ else } Q]]$
(Get0)	$t = 0$	$l[[a^{\Delta t}?(u) \text{ then } P \text{ else } Q]] \xrightarrow{\Delta 0@l} l[[Q]]$
(IfT)	$test = true$	$l[[\text{if } test \text{ then } P \text{ else } Q]] \xrightarrow{true@l} l[[P]]$
(IfF)	$test = false$	$l[[\text{if } test \text{ then } P \text{ else } Q]] \xrightarrow{false@l} l[[Q]]$
(DCall)	$t > 0$ and $id(u) \overset{def}{=} P_{id}$	$l[[\{v/u\}P_{id}]] \overset{t}{\rightsquigarrow} l[[P'_{id}]]$ implies $l[[id(v)]] \overset{t}{\rightsquigarrow} l[[P'_{id}]]$
(Call)	$id(u) \overset{def}{=} P_{id}$	$l[[\{v/u\}P_{id}]] \xrightarrow{call@l} l[[P'_{id}]]$ implies $l[[id(v)]] \xrightarrow{call@l} l[[P'_{id}]]$
(DPar)	$t > 0$	$N_1 \overset{t}{\rightsquigarrow} N'_1$, $N_2 \overset{t}{\rightsquigarrow} N'_2$ and $N_1 \mid N_2 \overset{\lambda}{\not\rightarrow}$ implies $N_1 \mid N_2 \overset{t}{\rightsquigarrow} N'_1 \mid N'_2$
(Par)		$N_1 \xrightarrow{\lambda} N'_1$ implies $N_1 \mid N_2 \xrightarrow{\lambda} N'_1 \mid N_2$
(DEquiv)	$t > 0$	$N \equiv N', N' \overset{t}{\rightsquigarrow} N''$ and $N'' \equiv N'''$ implies $N \overset{t}{\rightsquigarrow} N'''$
(Equiv)		$N \equiv N', N' \xrightarrow{\lambda} N''$ and $N'' \equiv N'''$ implies $N \xrightarrow{\lambda} N'''$

Rule (Move0) models an agent $go^{\Delta 0} l'$ then P at location l, where a migration action with timer 0 allows the agent to move to location l' and continue as P. Rule (Com) models the interaction of two co-located agents $a^{\Delta t}!\langle v\rangle$ then P else Q and $a^{\Delta t'}?(u)$ then P' else Q' on channel a, where the value v send by the first agent is used by the other agent in order to instantiate in P' the variable u. Note that during the communication phase the two agents stay in the same location and continue executing as P and $\{v/u\}P'$, respectively. Rules (Put0) and (Get0) model agents $a^{\Delta 0}!\langle v\rangle$ then P else Q and $a^{\Delta 0}?(u)$ then P else Q at location l, where a communication action timer expires, causing the agent to drop the action and continue as Q at its current location.

Rules (IfT) and (IfF) describe an agent if $test$ then P else Q that evaluates $test$ to decide whether to continue locally as P or as Q. Rule (Call) models an agent $id(v)$ being unfolded according to the definition of id, where the value v is used to instantiate the variable u. Rules (Equiv) and (DEquiv) describe replacing a reactive system with an equivalent one under the congruence relation $\equiv$. Rule (Par) models the construction of larger reactive systems by composing smaller ones in parallel.

The passage of time is modelled using the rules prefixed with D in Table 3. In rule (DPar),

the notation $N_1 \mid N_2 \stackrel{\lambda}{\not\rightarrow}$ indicates that there is no action λ and no reactive system $N_1' \mid N_2'$ such that $N_1 \mid N_2 \stackrel{\lambda}{\rightarrow} N_1' \mid N_2'$ can be derived using rules not prefixed with D, while the real-valued timeout t indicates that during this period none of the rules not prefixed with D can be applied. Also, the negative premise $N_1 \mid N_2 \stackrel{\lambda}{\not\rightarrow}$ is used to separate action execution from time progression, while ensuring consistency of the rule set [13].

The rules of Table 3 express execution of individual actions. For a multiset of actions $\Lambda = \{\lambda_1, \ldots, \lambda_m\}$, the notation $N \stackrel{\Lambda}{\rightarrow} N'$ is also used to capture a derivation of the form $N \stackrel{\lambda_1}{\longrightarrow} \ldots \stackrel{\lambda_m}{\longrightarrow} N'$. If $\Lambda = \{\lambda\}$, then the transition $N \stackrel{\{\lambda\}}{\longrightarrow} N'$ is written as $N \stackrel{\lambda}{\rightarrow} N'$.

A complete computational step is defined as the execution of a multiset of actions Λ, succeeded by a time step of length t:

$$N \stackrel{\Lambda}{\rightarrow} N_1 \stackrel{t}{\rightsquigarrow} N'.$$

The following result shows that if a reactive system permits only time to pass, the resulting behaviour is deterministic (the system obtained is unique), and time progresses continuously without skipping instances for executing migration or communication actions.

Proposition 1. *Consider a reactive system N such that $N \stackrel{t}{\rightsquigarrow} N'$.*

1. *If exists a reactive system N'' such that $N \stackrel{t}{\rightsquigarrow} N''$ then $N' \equiv N''$;*
2. *It exists a reactive system N'' such that $N \stackrel{t''}{\rightsquigarrow} N''$ and $N'' \stackrel{t'}{\rightsquigarrow} N'$ iff $t = t' + t''$.*

Proof. 1. The induction method depends on the structure of the reactive system N.

- If $N = l[[\mathbf{0}]]$, then from Table 3 the only time passing rule applicable to the reactive system N is (DSTOP). This means that if $N \stackrel{t}{\rightsquigarrow} N'$ then $N' = l[[\mathbf{0}]]$, and also if $N \stackrel{t}{\rightsquigarrow} N''$ then $N'' = l[[\mathbf{0}]]$. Thus, $N' = N''$ and $N' \equiv N''$.
- If $N = l[[a^{\Delta t'}!\langle v\rangle \text{ then } P \text{ else } Q]]$, then from Table 3 the only time passing rule applicable to the reactive system N is (DPUT). This means that if $N \stackrel{t}{\rightsquigarrow} N'$ then $N' = l[[a^{\Delta t'-t}!\langle v\rangle \text{ then } P \text{ else } Q]]$, and also if $N \stackrel{t}{\rightsquigarrow} N''$ then $N'' = l[[a^{\Delta t'-t}!\langle v\rangle \text{ then } P \text{ else } Q]]$. Thus, $N' = N''$ and $N' \equiv N''$.
- If $N = l[[a^{\Delta t'}?(u) \text{ then } P \text{ else } Q]]$, then from Table 3 the only time passing rule applicable to the reactive system N is (DGET). This means that if $N \stackrel{t}{\rightsquigarrow} N'$ then $N' = l[[a^{\Delta t'-t}?(u) \text{ then } P \text{ else } Q]]$, and also if $N \stackrel{t}{\rightsquigarrow} N''$ then $N'' = l[[a^{\Delta t'-t}?(u) \text{ then } P \text{ else } Q]]$. Thus, $N' = N''$ and $N' \equiv N''$.
- If $N = l[[go^{\Delta t'} l' \text{ then } P]]$ then from Table 3 the only time passing rule applicable to the reactive system N is (DMOVE). This means that if $N \stackrel{t}{\rightsquigarrow} N'$ then $N' = l[[go^{\Delta t'-t} l' \text{ then } P]]$, and also if $N \stackrel{t}{\rightsquigarrow} N''$ then $N'' = l[[go^{\Delta t'-t} l' \text{ then } P]]$. Thus, $N' = N''$ and $N' \equiv N''$.
- If $N = N_1 \mid N_2$, then from Table 3 the only time passing rule applicable to the reactive system N is (DPAR). This means that if $N \stackrel{t}{\rightsquigarrow} N'$ then $N' = N_1' \mid N_2'$, where $N_1 \stackrel{t}{\rightsquigarrow} N_1'$ and $N_2 \stackrel{t}{\rightsquigarrow} N_2'$, and also if $N \stackrel{t}{\rightsquigarrow} N''$ then $N'' = N_1'' \mid N_2''$, where $N_1 \stackrel{t}{\rightsquigarrow} N_1''$ and $N_2 \stackrel{t}{\rightsquigarrow} N_2''$. By induction, since $N_1 \stackrel{t}{\rightsquigarrow} N_1'$ and $N_1 \stackrel{t}{\rightsquigarrow} N_1''$ it holds that $N_1' \equiv N_1''$, and also since $N_2 \stackrel{t}{\rightsquigarrow} N_2'$ and $N_2 \stackrel{t}{\rightsquigarrow} N_2''$ it holds that $N_2' \equiv N_2''$. Thus, $N_1' \mid N_2' \equiv N_1'' \mid N_2''$ and $N' \equiv N''$.

2. The induction method used depends on the structure of the reactive system N.

   - If $N = l[[\mathbf{0}]]$, then from Table 3 the only time passing rule applicable to the reactive system N is (DSTOP). This means that if $N \overset{t}{\rightsquigarrow} N''$ then $N'' = l[[\mathbf{0}]]$, and also if $N'' \overset{t}{\rightsquigarrow} N'$ then $N' = l[[\mathbf{0}]]$. Since $N = N' = l[[\mathbf{0}]]$, then rule (DSTOP) can be used with time passing for $t + t'$ time units, namely $N \overset{t+t'}{\rightsquigarrow} N'$.
   - If $N = l[[a^{\Delta t''}!\langle v\rangle \text{ then } P \text{ else } Q]]$, then from Table 3 the only time passing rule applicable to the reactive system N is (DPUT). This means that if $N \overset{t}{\rightsquigarrow} N''$ then $N'' = l[[a^{\Delta t''-t}!\langle v\rangle \text{ then } P \text{ else } Q]]$, with $t'' \geq t \geq 0$, and also if $N'' \overset{t}{\rightsquigarrow} N'$ then $N' = l[[a^{\Delta(t''-t)-t'}!\langle v\rangle \text{ then } P \text{ else } Q]]$, with $t'' \geq t + t' \geq 0$. Thus, the rule (DPUT) can be used with time passing for $t + t'$ time units, namely $N \overset{t+t'}{\rightsquigarrow} N'$.
   - If $N = l[[a^{\Delta t''}?(u) \text{ then } P \text{ else } Q]]$, then from Table 3 the only time passing rule applicable to the reactive system N is (DGET). This means that if $N \overset{t}{\rightsquigarrow} N''$ then $N'' = l[[a^{\Delta t''-t}?(u) \text{ then } P \text{ else } Q]]$, with $t'' \geq t \geq 0$, and also if $N'' \overset{t}{\rightsquigarrow} N'$ then $N' = l[[a^{\Delta(t''-t)-t'}?(u) \text{ then } P \text{ else } Q]]$, with $t'' \geq t + t' \geq 0$. Thus, the rule (DGET) can be used with time passing for $t + t'$ time units, namely $N \overset{t+t'}{\rightsquigarrow} N'$.
   - If $N = l[[go^{\Delta t''} l' \text{ then } P]]$, then from Table 3 the only time passing rule applicable to the reactive system N is (DMOVE). This means that if $N \overset{t}{\rightsquigarrow} N''$ then $N'' = l[[go^{\Delta t''-t} l' \text{ then } P]]$, with $t'' \geq t \geq 0$, and also if $N'' \overset{t}{\rightsquigarrow} N'$ then $N' = l[[go^{\Delta(t''-t)-t'} l' \text{ then } P]]$, with $t'' \geq t + t' \geq 0$. Thus, the rule (DMOVE) can be used with time passing for $t + t'$ time units, namely $N \overset{t+t'}{\rightsquigarrow} N'$.
   - If $N = N_1 \mid N_2$, then from Table 3 the only time passing rule applicable to the reactive system N is (DPAR). This means that if $N \overset{t}{\rightsquigarrow} N''$ then $N \overset{\lambda}{\not\rightarrow}$ and $N'' = N_1'' \mid N_2''$, where $N_1 \overset{t}{\rightsquigarrow} N_1''$ and $N_2 \overset{t}{\rightsquigarrow} N_2''$, and also if $N'' \overset{t'}{\rightsquigarrow} N'$ then $N'' \overset{\lambda}{\not\rightarrow}$ and $N' = N_1' \mid N_2'$, where $N_1'' \overset{t'}{\rightsquigarrow} N_1'$ and $N_2'' \overset{t'}{\rightsquigarrow} N_2'$. By induction, since $N_1 \overset{t}{\rightsquigarrow} N_1''$ and $N_1'' \overset{t'}{\rightsquigarrow} N_1'$ it holds that $N_1 \overset{t+t'}{\rightsquigarrow} N_1'$, and also since $N_2 \overset{t}{\rightsquigarrow} N_2''$ and $N_2'' \overset{t'}{\rightsquigarrow} N_2'$ it holds that $N_2 \overset{t+t'}{\rightsquigarrow} N_2'$. Since $N_1 \overset{t+t'}{\rightsquigarrow} N_1'$, $N_2 \overset{t+t'}{\rightsquigarrow} N_2'$ and $N \overset{\lambda}{\not\rightarrow}$, then rule (DPAR) can be used with time passing for $t + t'$ time units, namely $N \overset{t+t'}{\rightsquigarrow} N'$.

□

Example 2. *The resource example from [4], illustrated in Figure 1, is examined. The reactive system includes four places connected by roads. Each road has an associated cost that the driver must pay before traversing it (for instance, the road from location a to b costs $\$70$). In addition, the driver can use in some locations cash machines to withdraw money (e.g., $\$100$ at a and c).*

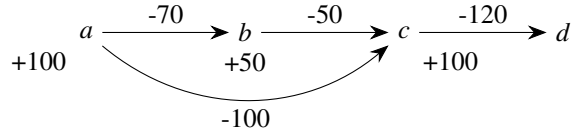

Fig. 1. A quantitative road system

If the driver starts at location a without money, it may withdraw $\$100$ and then choose to move either to b or c. Passing through b enables the driver to reach location d, whereas moving directly from a to c leaves the driver unable to proceed to d because of insufficient funds.

Next, the reactive system of Figure 1 is illustrated in rTIMO. *An agent $A(v)$ residing in a location l is assumed to allow a driver to use a channel w to withdraw $\$v$, after which the agent $A(v)$ is available to interact with other agents.*

$A(v) = w^{\Delta\infty}!\langle v\rangle$ then $A(v)?$ else $\mathbf{0}$.

and an agent $R(l', v)$ modelling the fact that there exists a road with cost v between the current location and l'; the agent $R(l', v)$ uses the channel r to communicate the cost and target location to a willing driver:

$R(l, v) = r\,^{\Delta\infty}!\langle l, v\rangle$ then $R(l, v)$ else $\mathbf{0}$.

A driver with an initial amount $\$b$ of cash, who can use channels w and r to obtain cash and target locations, respectively, is defined as:

$$D(b) = w^{\Delta\infty}?(x) \text{ then } r\,^{\Delta\infty}?(y, z) \text{ then if } b + x \geq z \text{ then go}\,^{\Delta 2}y \text{ then } D(b + x - z) \text{ else } D'(b + x) \text{ else } \mathbf{0} \text{ else } \mathbf{0}$$

where $D'(b + x)$ indicates that the driver lacks sufficient funds to proceed.

In summary, the initial reactive system shown in Figure 1 is modelled in rTIMO *as:*

$$E = a[[A(100) \mid R(b, 70) \mid R(c, 100) \mid D(0)]] \mid b[[A(50) \mid R(c, 50)]] \mid c[[A(100) \mid R(d, 120)]] \mid d[[\,]].$$

From the rules from Table 3, two possible reactive systems can be derived. In the first, once 2 time units have passed, the D reaches location c and is unable to continue to location d due to lack of funds:

$$a[[A(100) \mid R(b, 70) \mid R(c, 100)]] \mid b[[A(50) \mid R(c, 50)]] \mid c[[A(100) \mid R(d, 120) \mid D'(100)]] \mid d[[\,]]\,,$$

and once 6 time units have passed, where the D arrives at location d:

$$a[[A(100) \mid R(b, 70) \mid R(c, 100)]] \mid b[[A(50) \mid R(c, 50)]] \mid c[[A(100) \mid R(d, 120)]] \mid d[[D(10)]].$$

3. Behavioural Equivalences in Reactive Systems

Bisimulations are an important method for analyzing and comparing the behaviour of distributed systems [14]. A weaker form of bisimulation is adopted in which two reactive systems are regarded as equivalent if, at each computational step, they can execute the same multiset of actions (independent of order), and subsequently they can execute either a communication or a time step. This contrasts with the classical approach, which considers two systems equivalent only if they perform the same sequence of actions with time steps of equal duration, as in the π-calculus [1] and ω-regular languages [15].

The choice of a behavioural equivalence relation is determined by the reactive system under study and the properties that must be kept unchanged. An advantage of defining equivalence relations compositional with respect to the main constructs of rTIMO is that a reactive system can be replaced with an equivalent one but no unintended side effects are introduced.

Building on the approaches in reconfigurable systems [16], TIMO [17] and timed distributed π-calculus [18], the standard notion of bisimilarity is expanded to explicitly include communicated values and instantiated variables. Before introducing the bisimulations for reactive systems,

some useful basic notions are first recalled. For a set $\mathcal{N}$ of reactive systems, the identity relation over it is defined as

$$\mathrm{id} \stackrel{def}{=} \{(N, N) \mid N \in \mathcal{N}\}. \quad (1)$$

Given a relation $\mathcal{R}$, its inverse is defined as

$$\mathcal{R}^{-1} \stackrel{def}{=} \{(N_2, N_1) \mid (N_1, N_2) \in \mathcal{R}\}. \quad (2)$$

The composition of two relations $\mathcal{R}_1$ and $\mathcal{R}_2$ is defined as the relation

$$\mathcal{R}_1\mathcal{R}_2 \stackrel{def}{=} \{(N, N'') \mid \exists N' \in \mathcal{N} \text{ s.t. } (N, N') \in \mathcal{R}_1 \text{ and } (N', N'') \in \mathcal{R}_2\}. \quad (3)$$

From this point, the notation $\Sigma = \bigcup_{l,l' \in Loc} \{l \rhd l', call@l, \Delta 0@l, true@l, false@l\}$ is used for the set of labels appearing in Table 3 excluding the communication labels.

Definition 3. *A symmetric binary relation $\mathcal{R}$ over $\mathcal{N}$ is a timed communication bisimulation (TC bisimulation) if, for every $(N, M) \in \mathcal{R}$, the following holds for every $\Lambda \in \Sigma^*$ and $t \in \mathbb{R}_+$:*

- *if $N \xrightarrow{\Lambda} \xrightarrow{\{v/u\}@l} N'$ then exists M' such that $M \xrightarrow{\Lambda} \xrightarrow{\{v/u\}@l} M'$ and $(N', M') \in \mathcal{R}$;*
- *if $N \xrightarrow{\Lambda} \stackrel{t}{\rightsquigarrow} N'$ then exists M' such that $M \xrightarrow{\Lambda} \stackrel{t}{\rightsquigarrow} M'$ and $(N', M') \in \mathcal{R}$.*

Two reactive systems N and M are said to be timed communication bisimilar, denoted by $N \sim M$, if $(N, M) \in \mathcal{R}$ for some TC bisimulation $\mathcal{R}$. Formally:

$$\sim = \bigcup \{\mathcal{R} \mid \mathcal{R} \text{ is a } TC \text{ } bisimulation\}.$$

In Definition 3, both action and timed transitions are handled uniformly; therefore, the TC bisimulation works as the classical bisimulation.

Proposition 4. *The largest TC bisimulation is the equivalence relation $\sim$.*

Proof. To establish that the relation $\sim$ is an equivalence, it is shown that it is reflexive ($N \sim N$), symmetric ($N \sim N'$ implies $N' \sim N$), and transitive ($N \sim N'$ and $N' \sim N''$ imply $N \sim N''$).

1. Reflexivity: Note that, $N \sim N$ holds only if there exists a binary relation $\mathcal{R}$ such that $(N, N) \in \mathcal{R}$ and $\mathcal{R}$ is a TC bisimulation. It is proven that id is such a relation. Since $(N, N) \in \mathcal{R}$, (1) implies that $(N, N) \in \mathrm{id}$.

   (i) if $N \xrightarrow{\Lambda} \xrightarrow{\{v/u\}@l} N'$, then $(N', N') \in \mathrm{id}$;

   (ii) if $N \xrightarrow{\Lambda} \stackrel{t}{\rightsquigarrow} N'$, then $(N', N') \in \mathrm{id}$.

   Thus, id is a TC bisimulation, and Definition 3 implies that if $(N, N) \in \mathrm{id}$ then $(N, N) \in \sim$, namely it results the desired relation $N \sim N$.

2. Symmetry: Assume $N \sim N'$; this implies that there exists some TC bisimulation $\mathcal{R}$ such that $(N, N') \in \mathcal{R}$. Note that, $N' \sim N$ holds only if there exists some TC bisimulation $\mathcal{R}'$ such that $(N', N) \in \mathcal{R}'$. It is proven that $\mathcal{R}^{-1}$ is such a relation. Since $(N, N') \in \mathcal{R}$, (2) implies that $(N', N) \in \mathcal{R}^{-1}$. As $\mathcal{R}$ is a TC bisimulation, it implies that:

   (i) if $N \xrightarrow{\Lambda} \xrightarrow{\{v/u\}@l} N_1$ then $\exists N_1'$, $N' \xrightarrow{\Lambda} \xrightarrow{\{v/u\}@l} N_1'$, $(N_1, N_1') \in \mathcal{R}$, namely it results that $(N_1', N_1) \in \mathcal{R}^{-1}$;

   (ii) if $N \xrightarrow{\Lambda} \stackrel{t}{\rightsquigarrow} N_1$ then $\exists N_1'$, $N' \xrightarrow{\Lambda} \stackrel{t}{\rightsquigarrow} N_1'$, $(N_1, N_1') \in \mathcal{R}$, namely it results that $(N_1', N_1) \in \mathcal{R}^{-1}$.

   Thus, $\mathcal{R}^{-1}$ is a TC bisimulation, and Definition 3 implies that $(N', N) \in \mathcal{R}^{-1}$ implies $(N', N) \in \sim$, namely it results the desired relation $N' \sim N$.

3. Transitivity: Assume $N \sim N'$ and $N' \sim N''$; this implies that there exists some TC bisimulation $\mathcal{R}_1$ such that $(N, N') \in \mathcal{R}_1$ and there exists some TC bisimulation $\mathcal{R}_2$ such that $(N', N'') \in \mathcal{R}_2$. Note that, $N \sim N''$ holds only if there exists some TC bisimulation $\mathcal{R}'$ such that $(N, N'') \in \mathcal{R}'$ and $\mathcal{R}'$. It is proven that $\mathcal{R}_1\mathcal{R}_2$ is such a relation. Since $(N, N') \in \mathcal{R}_1$ and $(N', N'') \in \mathcal{R}_2$, (3) implies that $(N, N'') \in \mathcal{R}_1\mathcal{R}_2$. As $\mathcal{R}_1$ and $\mathcal{R}_2$ are TC bisimulations, it implies that:

   (i) if $N \xrightarrow{\Lambda} \xrightarrow{\{v/u\}@l} N_1$ then $\exists N_1'$, $N' \xrightarrow{\Lambda} \xrightarrow{\{v/u\}@l} N_1'$, $(N_1, N_1') \in \mathcal{R}_1$, and thus also $\exists N_1''$, $N'' \xrightarrow{\Lambda} \xrightarrow{\{v/u\}@l} N_1''$, $(N_1', N_1'') \in \mathcal{R}_2$ namely it results that $(N_1, N_1'') \in \mathcal{R}_1\mathcal{R}_2$;

   (ii) if $N \xrightarrow{\Lambda} \overset{t}{\rightsquigarrow} N_1$ then $\exists N_1'$, $N' \xrightarrow{\Lambda} \overset{t}{\rightsquigarrow} N_1'$, $(N_1, N_1') \in \mathcal{R}_1$, and thus also $\exists N_1''$, $N'' \xrightarrow{\Lambda} \overset{t}{\rightsquigarrow} N_1''$, $(N_1', N_1'') \in \mathcal{R}_2$ namely it results that $(N_1, N_1'') \in \mathcal{R}_1\mathcal{R}_2$.

   Thus, $\mathcal{R}_1\mathcal{R}_2$ is a TC bisimulation, and Definition 3 implies that $(N, N'') \in \mathcal{R}_1\mathcal{R}_2$ implies $(N, N'') \in \sim$, namely it results the desired relation $N \sim N''$.

To establish that the largest TC bisimulation is $\sim$, it is first shown that $\sim$ is a TC bisimulation. Assume $(N, N') \in \sim$; this implies that there exists some TC bisimulation $\mathcal{R}$ such that $(N, N') \in \mathcal{R}$. As $\mathcal{R}$ is a TC bisimulation, it implies that:

(i) if $N \xrightarrow{\Lambda} \xrightarrow{\{v/u\}@l} N_1$ then $\exists N_1'$, $N' \xrightarrow{\Lambda} \xrightarrow{\{v/u\}@l} N_1'$, $(N_1, N_1') \in \mathcal{R}$, namely it results that $(N_1', N_1) \in \sim$;

(ii) if $N \xrightarrow{\Lambda} \overset{t}{\rightsquigarrow} N_1$ then $\exists N_1'$, $N' \xrightarrow{\Lambda} \overset{t}{\rightsquigarrow} N_1'$, $(N_1, N_1') \in \mathcal{R}$, namely it results that $(N_1', N_1) \in \sim$.

Thus $\sim$ is a TC bisimulation, and since it includes all TC bisimulations, then it holds that it is the largest TC bisimulation. □

An important result is that the TC bisimilarity is useful to compare the behaviour of reactive systems when considering complete computational steps.

Proposition 5. *If $N \sim M$, $N \xrightarrow{\Lambda} N' \overset{t}{\rightsquigarrow} N''$ and $M \xrightarrow{\Lambda} M' \overset{t}{\rightsquigarrow} M''$, then $N'' \sim M''$.*

Proof. Depending on the values contained in the finite multiset of actions Λ, Λ can be written as $\Lambda_1 \cup \{\lambda_1\} \cup \ldots \cup \Lambda_k$, where $\Lambda_i \in \Sigma^*$, for $1 \leq i \leq k$, and $\lambda_i \in \bigcup_{l \in Loc} \{\{v/u\}@l\}$, for $1 \leq i \leq k-1$. In this case, $N \xrightarrow{\Lambda} N' \overset{t}{\rightsquigarrow} N''$ can be rewritten as $N \xrightarrow{\Lambda_1} N_1' \xrightarrow{\lambda_1} N_1 \xrightarrow{\Lambda_2} \ldots \xrightarrow{\Lambda_k} N_k' = N' \overset{t}{\rightsquigarrow} N''$. Since $N \sim M$ and $N \xrightarrow{\Lambda_1}\xrightarrow{\lambda_1} N_1$, then using Definition 3 it holds that there exists a reactive system M_1 such that $M \xrightarrow{\Lambda_1}\xrightarrow{\lambda_1} M_1$ and $N_1 \sim M_1$. By induction on the number of steps performed, it holds that since $N_1 \sim M_i$ and $N_i \xrightarrow{\Lambda_{i+1}}\xrightarrow{\lambda_{i+1}} N_{i+1}$, then using Definition 3 there exists a reactive system M_{i+1} such that $M_i \xrightarrow{\Lambda_{i+1}}\xrightarrow{\lambda_{i+1}} M_{i+1}$ and $N_{i+1} \sim M_{i+1}$, for $1 \leq i \leq k-2$. Similarly, since $N_{k-1} \sim M_{k-1}$, $N_{k-1} \xrightarrow{\Lambda_k} \overset{t}{\rightsquigarrow} N''$ and $M_{k-1} \xrightarrow{\Lambda_k} \overset{t}{\rightsquigarrow} M''$, then using Definition 3 it results the desired relation $N'' \sim M''$. □

Definition 3 addresses the evolution of entire reactive systems, but it does not take into account the parallel composition of systems. For instance, consider two reactive systems:

$$N_1 = l[[a^{\Delta\infty}!\langle 10\rangle \text{ then } \mathbf{0}]] \text{ and } N_2 = l[[b^{\Delta\infty}?(x) \text{ then } \mathbf{0}]]$$

Clearly, $N_1 \sim N_2$, since both reactive systems permit only transitions of the form $\overset{t}{\rightsquigarrow}$. However, when composed with $N = l[[a^{\Delta\infty}?(x)\text{then } \mathbf{0}]]$, it is obtained $N_1 \mid N \not\sim N_2 \mid N$ since the first system can perform the transition $\xrightarrow{10/x@l}$, while the second can only perform $\overset{t}{\rightsquigarrow}$.

Demanding an exact correspondence between the executed multiset of actions followed by either a communication or a time step, can be overly restrictive for TC equivalences. Since in Example 2 the observables are the costs, a bisimilarity is next defined that considers as equivalent two reactive systems that are able to communicate the same values.

Definition 6. *A symmetric binary relation $\mathcal{R}$ over $\mathcal{N}$ is a timed resource bisimulation (TR bisimulation) if, for every $(N, M) \in \mathcal{R}$, the following holds for every $\Lambda \in \Sigma^*$ and $t \in \mathbb{R}_+$:*

- *if $N \xrightarrow{\Lambda} \xrightarrow{\{v/u\}@l} N'$ then exists u',M' such that $M \xrightarrow{\Lambda} \xrightarrow{\{v/u'\}@l} M'$ and $(N', M') \in \mathcal{R}$;*
- *if $N \xrightarrow{\Lambda} \overset{t}{\rightsquigarrow} N'$ then exists t', M', such that $M \xrightarrow{\Lambda} \overset{t'}{\rightsquigarrow} M'$ and $(N', M') \in \mathcal{R}$.*

Two reactive systems N and M are said to be timed resource bisimilar, denoted by $N \approx M$, if $(N, M) \in \mathcal{R}$ for some TR bisimulation $\mathcal{R}$. Formally:

$$\approx = \bigcup\{\mathcal{R} \mid \mathcal{R}\ is\ a\ TR\ bisimulation\}.$$

Proposition 7. *The largest TR bisimulation is the equivalence relation $\approx$.*

Proof. The proof proceeds in a similar manner as for Proposition 4. □

Analogous to TC bisimilarity, TR bisimilarity is useful to compare the behaviour of reactive systems when considering complete computational steps.

Proposition 8. *If $N \approx M$, $N \xrightarrow{\Lambda} N' \overset{t}{\rightsquigarrow} N''$ and $M \xrightarrow{\Lambda} M' \overset{t'}{\rightsquigarrow} M''$, then $N'' \approx M''$.*

Proof. The proof proceeds in a similar manner as for Proposition 5. □

Example 9. *Building on the reactive system E, consider a variant, denoted E_1, which includes a slightly modified version of the D, defined as follows:*

$D_1(b) = w^{\Delta\infty}?(x_1)$ then $r^{\Delta\infty}?(y_1, z_1)$ then if $b + x_1 \geq z_1$ then go$^{\Delta 4} y_1$ then $D_1(b + x_1 - z_1)$
else $D_1'(b + x_1)$
else $\mathbf{0}$
else $\mathbf{0}$.

Since D_1 employs different variables for communication, then $E \xrightarrow{100/x@l} E'$ and $E_1 \xrightarrow{100/x_1@l} E_1'$, which implies $E \not\sim E_1$. However, after execution it holds that $E \approx E_1$.

Since $\sim$ is stronger than $\approx$, the following result holds.

Proposition 10. *If N and N' are TC bisimilar, they are also TR bisimilar.*

Proof. Since $N \sim N'$, this implies that there exists some TC bisimulation $\mathcal{R}$ such that $(N, N') \in \mathcal{R}$. Note that, $N \approx N'$ holds only if there exists some TC bisimulation $\mathcal{R}'$ such that $(N, N') \in \mathcal{R}'$. Assume $(N, N') \in \mathcal{R}'$. As $\mathcal{R}$ is a TC bisimulation, it implies that:

(i) if $N \xrightarrow{\Lambda} \xrightarrow{\{v/u\}@l} N_1$ then $\exists N_1'$, $N' \xrightarrow{\Lambda} \xrightarrow{\{v/u\}@l} N_1'$; by choosing $u' = u$, it holds that $\exists N_1'$, $N' \xrightarrow{\Lambda} \xrightarrow{\{v/u\}@l} N_1'$ can be rewritten as $\exists N_1', u'$, $N' \xrightarrow{\Lambda} \xrightarrow{\{v/u'\}@l} N_1'$, and by Definition 6 it results that $(N_1, N_1') \in \mathcal{R}'$;

(ii) if $N \xrightarrow{\Lambda} \overset{t}{\rightsquigarrow} N_1$ then $\exists N_1'$, $N' \xrightarrow{\Lambda} \overset{t}{\rightsquigarrow} N_1'$, and by Definition 6 it results that $(N_1, N_1') \in \mathcal{R}'$.

Thus $\mathcal{R}'$ is a TR bisimulation, and Definition 6 implies that $(N, N') \in \mathcal{R}'$ implies $(N', N) \in \approx$, namely $N \approx N'$. □

The previous example shows that the reverse implication is not valid; that is, if N and N' are TR bisimilar, they are not necessarily TC bisimilar.

4. Conclusions

This paper enhances the modelling of reactive and reconfigurable systems by presenting the rTIMO calculus which integrates timed communication and migration in distributed computing. Timed communication and resource bisimulations extend traditional behavioural equivalence, enabling the description of complex timing and resource-sensitive interactions in mobile systems. The results reported in this paper establish properties ensuring compositional and realistic system analysis. Future research directions include extending this approach to incorporate probabilistic behaviours, security properties, and application to emerging multi-agent platforms.

This paper explores various aspects of modelling behaviours and interactions in multi-agent systems, providing a theoretical foundation for understanding reactive systems through the rTIMO calculus. This calculus facilitates reactive communication and mobility via channel and location exchanges. A complete computational step is defined as executing a multiset of actions before a time step occurs. The rTIMO calculus is derived from TIMO [9], linking theoretical process calculi with practical multi-agent languages [19]. Over time, an entire family of TIMO variants has been introduced and studied, including PerTIMO [20] which addresses access permissions, and BigTiMo [21] which integrates bigraphs of [22] with TIMO . Authors' current approach compares the behaviours of reactive systems by accounting for the actions performed and their corresponding timers.

Authors' current approach compares the behaviours of reactive systems by considering actions performed and their corresponding timers. By introducing new forms of bisimulation, one can regard as equivalent certain reactive systems that would otherwise be distinguished. In particular, the concepts of behavioural equivalences and spatial reconfiguration can significantly enhance the analysis of reactive systems, clarifying how system behaviour can adapt to changes. Authors' work on agent communication and navigation aims to inspire further investigations in real-world scenarios.

References

[1] R. MILNER, *Communicating and Mobile Systems: The π-calculus*, Cambridge University Press, New York, NY, USA, 1999.

[2] C. ARECES, R. FERVARI and G. HOFFMANN, *Relation-changing modal operators*, Logic Journal of the IGPL **23**(4), 2015, pp. 601–627.

[3] J. VAN BENTHEM, *An essay on sabotage and obstruction*, in Mechanizing Mathematical Reasoning, Essays in Honor of Jörg H. Siekmann on the Occasion of His 60th Birthday, D. Hutter and W. Stephan, Eds., Springer-Verlag, Berlin, Heidelberg, Lecture Notes in Computer Science **2605**, 2005, pp. 268–276.

[4] D. M. GABBAY, *Reactive Kripke Semantics*, Cognitive Technologies, Springer-Verlag, Berlin, Heidelberg, 2013.

[5] D. HAREL and A. PNUELI, *On the development of reactive systems*, in Logics and Models of Concurrent Systems - Conference proceedings, Colle-sur-Loup (near Nice), France, K. R. Apt, Ed., Springer, Berlin, Heidelberg, NATO ASI Series **13**, 1984, pp. 477–498.

[6] R. H. N. SANTIAGO, M. A. MARTINS and D. FIGUEIREDO, *Introducing fuzzy reactive graphs: a simple application on biology*, Soft Computing **25**(9), 2021, pp. 6759–6774.

[7] R. ALUR and D. L. DILL, *A theory of timed automata*, Theoretical Computer Science **126**(2), 1994, pp. 183–235.

[8] G. CIOBANU and C. PRISACARIU, *Timers for distributed systems*, in Proceedings QAPL 2006, Vienna, Austria, A. D. Pierro and H. Wiklicky, Eds., Elsevier, Electronic Notes in Theoretical Computer Science **164**, 2006, pp. 81–99.

[9] G. CIOBANU and M. KOUTNY, *Modelling and verification of timed interaction and migration*, in Proceedings FASE 2008, Budapest, Hungary, J. L. Fiadeiro and P. Inverardi, Eds., Springer, Berlin, Heidelberg, Lecture Notes in Computer Science **4961**, 2008, pp. 215–229.

[10] B. AMAN and G. CIOBANU, *Real-time migration properties of rTiMo verified in Uppaal*, in Proceedings SEFM 2013, Madrid, Spain, R. M. Hierons, M. G. Merayo and M. Bravetti, Eds., Springer, Berlin, Heidelberg, Lecture Notes in Computer Science **8137**, 2013, pp. 31–45.

[11] B. AMAN and G. CIOBANU, *Verification of critical systems described in real-time TiMo*, International Journal on Software Tools for Technology Transfer **19**(4), 2017, pp. 395–408.

[12] E. POSSE and J. DINGEL, *Kiltera: A language for timed, event-driven, mobile and distributed simulation*, in Proceedings DS-RT '10, Fairfax, VA, USA, S.J. Turner and D. J. Roberts, Eds., IEEE Computer Society, 2010, pp. 87–96.

[13] J. F. GROOTE, *Transition system specifications with negative premises*, Theoretical Computer Science **118**(2), 1993, pp. 263–299.

[14] D. SANGIORGI, *Introduction to Bisimulation and Coinduction*, Cambridge University Press, New York, NY, USA, 2011.

[15] D. M. R. PARK, *Concurrency and automata on infinite sequences*, in Theoretical Computer Science, 5th GI-Conference, Karlsruhe, Germany, P. Deussen, Ed., Springer, Berlin, Heidelberg, Lecture Notes in Computer Science **104**, 1981, pp. 167–183.

[16] B. AMAN and G. CIOBANU, *Behavioural Equivalences over Reconfigurable Systems*, in Proceedings ReacTS 2024, Aveiro, Portugal, J. Proença, R. Fervari, M. A. Martins, R. Kahle and G. Pluck, Eds., Springer, Berlin, Heidelberg, Lecture Notes in Computer Science **15551**, 2024, pp. 7–21.

[17] B. AMAN, G. CIOBANU and M. KOUTNY, *Behavioural equivalences over migrating processes with timers*, in Proceedings FMOODS/FORTE 2012, Stockholm, Sweden, H. Giese and G. Rosu, Eds., Springer, Berlin, Heidelberg, Lecture Notes in Computer Science **7273**, 2012, pp. 52–66.

[18] G. CIOBANU, *Behaviour equivalences in timed distributed pi-calculus*, in Software-Intensive Systems and New Computing Paradigms - Challenges and Visions, M. Wirsing, J. Banâtre, M. M. Hölzl and A. Rauschmayer, Eds. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science **5380**, 2008, pp. 190–208.

[19] G. CIOBANU and C. JURAVLE, *Flexible software architecture and language for mobile agents*, Concurrency and Computation: Practice and Experience **24**(6), 2012, pp. 559–571.

[20] G. CIOBANU and M. KOUTNY, *Timed migration and interaction with access permissions*, in Proceedings FM 2011, Limerick, Ireland, M. J. Butler and W. Schulte, Eds., Springer, Berlin, Heidelberg, Lecture Notes in Computer Science **6664**, 2011, pp. 293–307.

[21] W. XIE, H. ZHU, M. ZHANG, G. LU and Y. FANG, *Formalization and verification of mobile systems calculus using the rewriting engine Maude*, in Proceedings COMPSAC 2018, Tokyo, Japan, S. Reisman, S. I. Ahamed, C. Demartini, T. M. Conte, L. Liu, W. R. Claycomb, M. Nakamura, E. Tovar, S. Cimato, C. Lung, H. Takakura, J. Yang, T. Akiyama, Z. Zhang and K. Hasan, Eds, IEEE Computer Society, 2018, pp. 213–218.

[22] R. MILNER, *The Space and Motion of Communicating Agents*, Cambridge University Press, New York, NY, USA, 2009.