Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
ELSEVIER
Electronic Notes in Theoretical Computer Science 169 (2007) 133–146
www.elsevier.com/locate/entcs

Multi Labelled Transition Systems: A Semantic Framework for Nominal Calculi

Rocco De Nicola and Michele Loreti

Dipartimento di Sistemi e Informatica, Università di Firenze.
{denicola,loreti}@dsi.unifi.it

Abstract

Action Labelled transition systems (LTS) have proved to be a fundamental model for describing and proving properties of concurrent systems. In this paper,Multiple Labelled Transition Systems (MLTS) are introduced as generalizations of LTS that permit dealing also with systems features that are becoming more and more important when considering languages and models for network aware programming. MLTS permit describing not only the actions systems can perform but also system's resources usage and their handling (creation, revelation ...) of names. To show adeguacy of our proposal we show how MLTS can be used to describe the operational semantics of one of the most studied calculus for mobility: the asynchronous π-calculus.

Keywords: Mobile Code Languages, Temporal Logics of Programs, Coordination Models, Proof Systems

1 Introduction

A successful approach to the description of the behaviour of concurrent processes is the one based on *Labelled Transition Systems* (LTS). An LTS consists of a set of state $\mathcal{S}$, a set of transition labels $\mathcal{L}$ and a transition relation $\rightarrow \subseteq \mathcal{S} \times \mathcal{L} \times \mathcal{S}$. States correspond to the possible configurations a system can reach. Labels describe the actions a system can perform to interact with the environment. Transition relations model how a system, in a given configuration, evolves after the execution of an action.

For the concrete case of a, by now, classical process description language like CCS [13], we have that $\mathcal{S}$ is the set of CCS terms (or agent expressions); $\mathcal{L}$ consists of the input/output actions over channels that CCS processes can perform plus the distinct action τ; $\rightarrow$ describes the systems evolutions as determined by the execution of specific actions.

* The work presented in this paper has been partially supported by EU Project Software Engineering for Service-Oriented Overlay Computers (SENSORIA, contract IST-3-016004-IP-09).

doi:10.1016/j.entcs.2007.05.019

The simplicity and versatility of Labelled Transition Systems has greatly contributed to the development and exploitation of different process description languages. Indeed, LTS have been the common semantic framework that has been the basis for defining common formal tools, like temporal logics and behavioural equivalences useful for reasoning and establishing properties of concurrent systems.

In the last decade, stimulated by new applications of network aware programming, several new formalisms based on process descripting languages but with new constructs for modelling network topology, name passing, resources usage and mobility have been proposed. Here, we just want to mention the Distributed Join-calculus [10], the Distributed π-calculus [11], the Ambient calculus [5], the Seal calculus [6], Nomadic Pict [16] and KLAIM [8].

The new primitives of these distributed nominal calculi, are such that not only actions, but also names and resources play a central role when modelling systems behaviour. This fact renders it more difficult to use classical LTS as semantic models. To capture some of the distinguishing features of the new calculi, such as *name scoping* or *process and data distribution*, transition labels have to be *enriched* to carry information not only about the performed actions but also about the state of the system. This interplay has rendered more difficult the development of a general approach to distributed nominal calculi and we have witnessed to the development of a number of theories tailored on specific calculi that cannot be easily generalised. Indeed, each calculus requires its own set of transition labels that depends on its specific aspect. The drawback of having a variety of transition labels limits the possibility of using the operational semantics as the basis for defining common formal tools, like logics and equivalences.

We feel that it would be important to come up with a general operational model that, for distributed nominal calculi, can play the same role LTS have played for process description languages. Such operational model should permit naturally capturing all main aspects of the new class of systems and should represent the natural basis for interpreting temporal logics for describing systems properties.

As a candidate general operational model, we propose *Multiple Labelled Transition Systems* (MLTS). Our starting point is the consideration that a central role in the new set of mobile calculi is played by *names* and *resources*. Names can be *public*, i.e. known by the environment where the system is executing, or *private*; and systems, during execution, can *discover* some of the private names. Resources, instead, can be *used*, *consumed*, and *created*. Thus, the description of the behaviour of terms of mobile calculi has to take into account both names and resources. We have that *states* of MLTS describe systems configurations while their *naming structures* permit associating public (known by the environment) names to each state. *Resources* model data (e.g. the values exchanged over a channel), computational environments (e.g. the locations where processes can be executed) and network links (that can be used for interaction). Moreover, MLTS are equipped with different transition relations that capture the different aspects of systems behaviour namely *actions execution*, *resources creation and consumption* and *name revelation*.

Indeed, an MLTS consists of a set of *states*, a set of *resources*, a set of *transi-*

tion labels, a *naming structure* and *three* transition relations. The three transition relations describe systems interactions with the environment:

- the *computation relation* describes the interaction with the environment;
- the *resource relation* describes resource usage;
- the *revelation relation* describes the names revealed to the environment.

In the rest of the paper we shall first present our new model, then we will show usefulness of our proposal by using it to describe the operational semantics of a well known formalism generally recognized as a minimal common denominator of calculi for mobility: the asynchronous π-calculus ($A\pi$) [3,12]. Finally, we shall touch the issue of behavioural equivalence for MLTS and will propose an equivalence that induces the same identification of those induced by asynchronous bisimulation for $A\pi$.

2 Multi Labelled Transition Systems

In this section we introduce our variant of LTS that will be used as a general framework for describing the semantics of mobile calculi. As mentioned in the introduction a key role will be played by *names* that can be used to refer to *channels* (as in the case of π-calculus [15]) or to *localities* (as in the case of locality-based calculi like KLAIM). These names can be *public*, i.e. known by the environment where the system is running, or *private*. The system, during its computation, can *discover* some private names.

First of all, we introduce a few basic definitions that permit dealing with structured sets of names.

Definition 2.1 *Let $\mathcal{N}$ be a set of names and $X \subseteq \mathcal{N}$, a* name substitution *is a function $\sigma : \mathcal{N} \to \mathcal{N}$, where:*

- $\{y_1/x_1, \ldots, y_n/x_n\}$ *denotes the substitution that maps x_i into y_i and that is the identity on the other names;*
- $\sigma_1 \cdot \sigma_2$ *is the composition of σ_1 and σ_2;*
- $\sigma_1 \backslash X$ *is the substitution σ_2 such that:*

$$\sigma_2(n) = \begin{cases} n & \text{if } n \in X \\ \sigma_1(n) & \text{otherwise} \end{cases}$$

- $\emptyset$ *denotes the identity substitution;*
- Σ *denotes the set of functions (substitutions) $\mathcal{N} \to \mathcal{N}$.*

Definition 2.2 *Let $X \subseteq \mathcal{N}$ and $\sigma \in \Sigma$, $\sigma(X)$ is the image of X with respect to σ, namely:*

$$\sigma(X) = \left\{x' | \exists x \in X : \sigma(x) = x'\right\}$$

Definition 2.3 *Let S be a set, a* naming structure *for S is a triple $\mathbb{N} = \langle \mathcal{N}, \eta, \circ \rangle$ such that:*

- $\mathcal{N}$ *is a countable set of names;*
- $\eta : S \to 2^{\mathcal{N}}$*, is the* naming function*;*
- $\circ : S \times \Sigma \to S$*, is the* renaming function*.*

where, if we use $s \circ \sigma$ *for* $\circ(s, \sigma)$*, we have that for each* $s \in S$ *and* $\sigma \in \Sigma$*:*

- $\eta(s)$ *is finite;*
- $s \circ \emptyset = s$*;*
- $\eta(s \circ \sigma) = \sigma(\eta(s))$*.*

Intuitively, a naming structure for S permits considering the elements of S as containers of names. For each element s in S, $\eta(s)$ gives the set of names that appear in s while the application of an element s to a substitution σ $(s \circ \sigma)$ yields the element of S obtained by renaming each name in s according to σ.

A Multi Labelled Transition System, MLTS for short, consists of a set of *states* $\mathcal{S}$, a set of *resources* $\mathcal{R}$, a set of *transition labels* $\mathcal{L}$ and a naming structure $\langle N, \eta, \sigma \rangle$ for each of $\mathcal{S}$, $\mathcal{R}$ and $\mathcal{L}$.

States describe the configurations a system can reach. The naming structure labels each state by a set of names. These are the names that are public (known by the environment) when the system is in a that state. Resources are the necessary prerequisites for system evolutions.

Transition labels typically identify the actions a system can perform to interact with the environment. We also assume that $\mathcal{L}$ contains a special distinct transition label τ denoting internal/silent evolution of a system.

The transition relations provide information about the actions systems can perform to interact with the environment; about the reaction of systems to creation or deletion of new *resources* and about the handling by systems of private and public names.

MLTS have three different transition relations:

- the *computation relation*, $\longrightarrow \mathcal{S} \times \mathcal{L} \times \mathcal{S}$;
- the *resource relation*, $\cdots\!\!\blacktriangleright \subseteq \mathcal{S} \times (\{\oplus, \ominus\} \times \mathcal{R}) \times \mathcal{S}$;
- the *revelation relation*, $\hookrightarrow \subseteq \mathcal{S} \times \mathcal{N} \times \mathcal{S}$.
- The *computation relation* plays the same role as the transition relation in an LTS; it describes the actions a system can perform to interact with the environment.
- The *resource relation* describes how a system evolves when resources are created or consumed. If r is a resource, we have that $s_1 \overset{\ominus r}{\cdots\!\!\blacktriangleright} s_2$ is possible if r is available at state s_1; after r is consumed state s_2 is reached. Similarly, we have $s_1 \overset{\oplus r}{\cdots\!\!\blacktriangleright} s_2$ if the environment can add resource r to s_1 leading the system to s_2.
- The *revelation relation* describes the capability of a system to reveal a private name to the environment.

To guarantee the correct management of names, we need to require some specific properties for the three transition relations, namely that they be preserved by name

permutations.

Definition 2.4 *A substitution σ is a* name permutation *if it is bijective.*

Definition 2.5 *Let $\mathbb{N} = \langle \mathcal{N}, \eta, \circ \rangle$ be a naming structure for $A_1, \ldots, A_n$, we say that a relation $\mathscr{R} \subseteq A_1 \times \ldots \times A_n$ is* preserved by name permutations *if and only if for each* name permutation *σ:*

$$(a_1, \ldots, a_n) \in \mathscr{R} \Leftrightarrow (a_1 \circ \sigma, \ldots, a_n \circ \sigma) \in \mathscr{R}$$

Definition 2.6 *A* Multi Labelled Transition System *is a 7-uple*

$$\mathcal{M} = \langle \mathcal{S}, \mathcal{R}, \mathcal{L}, \mathbb{N}, \longrightarrow, \dashrightarrow, \hookrightarrow \rangle$$

where:

- *$\mathcal{S}$, $\mathcal{R}$ and $\mathcal{L}$ are countable sets of* states, resources *and* transition labels*;*
- *$\mathbb{N} = \langle \mathcal{N}, \eta, \circ \rangle$ is a naming structure for $\mathcal{S}$, $\mathcal{R}$, $\mathcal{L}$ and $\mathcal{N}$; and*
- *the three relations*
  - $\longrightarrow \subseteq \mathcal{S} \times \mathcal{L} \times \mathcal{S}$*;*
  - $\dashrightarrow \subseteq \mathcal{S} \times (\{\oplus, \ominus\} \times \mathcal{R}) \times \mathcal{S}$*;*
  - $\hookrightarrow \subseteq \mathcal{S} \times \mathcal{N} \times \mathcal{S}$*;*

  are preserved by name permutations.

In the rest of the paper we will use $\mathcal{M}^{\mathcal{S}}$, $\mathcal{M}^{\mathcal{R}}$, $\mathcal{M}^{\mathcal{L}}$ and $\mathcal{M}^{\mathbb{N}}$ to denote, respectively, the set of state, the set of resources, the set of transition labels and the naming structure of MLTS $\mathcal{M}$.

3 Asynchronous π-Calculus

In this section we consider the asynchronous π-calculus ($A\pi$) [3,12], a subset of the π-calculus with no output prefixing and provide an MLTS for it.

First we define the syntax of $A\pi$. If Ch is a countable set of channels (whose elements are ranged over by a, b, ...), we let $\text{Proc}_{A\pi}$ be the set of $A\pi$ processes (P, Q,...) defined by the following syntax:

$$P, Q ::= \overline{a}b \mid G \mid P|Q \mid \nu a.P \mid !G$$
$$G ::= \mathbf{0} \mid a(b).P \mid \tau.P \mid G + G$$

where G denotes *guarded processes*.

In $A\pi$ processes interact by exchanging messages over channels.

The term $\overline{a}b$ is used to indicate the availability of message b over channel a; it models non-blocking outputs. Instead, the term $a(b).P$, indicates the process that retrieves a value over then channel a and then behaves like process P where the name b is bound to the retrieved value;

The term $\mathbf{0}$ denotes the deadlocked process and $\tau.P$ denotes the process that performs an internal action and then behaves like P; finally $G_1 + G_2$ indicates the

$$P|\mathbf{0} \equiv P \qquad P|Q \equiv Q|P \qquad (P|Q)|R \equiv P|(Q|R)$$

$$!P \equiv P|!P \qquad \frac{a \notin fn(Q)}{\nu\, a.(P|Q) \equiv (\nu\, a.P)|Q}$$

Table 1
$\textsc{Proc}_{A\pi}$ Structural Equivalence

process that can nondeterministically behave like G_1 or like G_2.

Processes are composed via *parallel composition*, $P|Q$ describe a system composed by two components (specified respectively by P and Q) that can proceed independently and that can interact via shared channels.

Private names are defined using *restriction*: $\nu a.P$ denotes that a is a private name in P.

Infinite behaviours are modelled via process *replication* ($!G$). This can be thought of as an infinite composition of P ($P|P|\ldots$).

Please notice that both input prefixing $b(a).P$ and name restriction $\nu a.P$ act as binders for name a within P. In the rest of the paper we shall use $fn(P)$ to denote the set of names free in P. We will also write $P =_\alpha Q$ to denote that P and Q are equals up-to renaming of bound names. For instance, $a(x).\overline{b}(x) =_\alpha a(y).\overline{b}(y)$ and $\nu\, a.\overline{b}a =_\alpha \nu\, c.\overline{b}c$. Let P be a process and σ a name substitution, we let $P \circ \sigma$ be the process obtained from P by replacing every free name a with $\sigma(a)$.

Terms that intuitively represent the same process are identified by means of standard *structural congruence* $\equiv$. This relation is defined as the smallest congruence relation over $A\pi$ processes induced by the laws in Table 1. The structural laws express that $|$ is commutative and associative and that the empty process can always be safely removed/added in a parallel composition. Structural equivalence states also that $!P$ is equivalent to $!P|P$ and that, if a does not occur in Q, $\nu\, a.(P|Q)$ is equivalent to $\nu\, a.P|Q$. In the sequel, elements in $\textsc{Proc}_{A\pi}$ will be considered equals up to structural equivalence.

Let us now consider a simple $A\pi$ specification for a simple system that we will use as a running example in the sequel.

Example 3.1 *A* proxy *is a system such that, given two channels a and b, if appropriate, emits the values read from channel a on channel b.*

A possible $A\pi$ specifications for this system is:

$$\textsc{Proxy}_1 = !a(x).\tau.\overline{b}x$$

where process $\textsc{Proxy}_1$ *models the system as the infinite replication of a process that first reads a value x from channel a, then performs some internal actions and finally emits x on channel b.*

In [1] the operational semantics of $A\pi$ processes is described by means of the LTS relation ($\longmapsto$) defined in Table 2 where it is assumed that $\longmapsto$ is closed with respect to the structural congruence relation $\equiv$; this means that:

$$\overline{a}b \stackrel{\overline{a}b}{\longmapsto} \mathbf{0} \qquad \tau.P \stackrel{\tau}{\longmapsto} P$$

$$\frac{P \stackrel{\overline{a}b}{\longmapsto} P' \quad a \neq b}{\nu\, b.P \stackrel{\overline{a}(b)}{\longmapsto} P'} \qquad a(b).P \stackrel{ac}{\longmapsto} P[c/b]$$

$$\frac{P \stackrel{\lambda}{\longmapsto} P' \quad a \notin n(\lambda)}{\nu\, a.P \stackrel{\lambda}{\longmapsto} \nu\, a.P'} \qquad \frac{G \stackrel{\lambda}{\longmapsto} P}{G + G' \stackrel{\lambda}{\longmapsto} P}$$

$$\frac{P \stackrel{\overline{a}b}{\longmapsto} P' \quad Q \stackrel{ab}{\longmapsto} Q'}{P|Q \stackrel{\tau}{\longmapsto} P'|Q'} \qquad \frac{P \stackrel{\overline{a}(b)}{\longmapsto} P' \quad Q \stackrel{ab}{\longmapsto} Q' \quad b \notin fn(Q)}{P|Q \stackrel{\tau}{\longmapsto} \nu\, b.P'|Q'}$$

$$\frac{P \stackrel{\lambda}{\longmapsto} P' \quad fn(Q) \cap bn(\alpha) = \emptyset}{P|Q \stackrel{\lambda}{\longmapsto} P'|Q'} \qquad \frac{G \stackrel{\lambda}{\longmapsto} P}{!G \stackrel{\lambda}{\longmapsto} P|!G}$$

Table 2
Labelled transition system for $A\pi$

$$\frac{P \equiv Q \quad Q \stackrel{\lambda}{\longmapsto} Q' \quad Q' \equiv P'}{P \stackrel{\lambda}{\longmapsto} P'}$$

As usual, bound and free names are considered and we have $n(\lambda) = fn(\lambda) \cup bn(\lambda)$ where:

$$fn(\tau) = \emptyset \;\; fn(\overline{a}(b)) = \{a\} \;\; fn(\overline{a}b) = fn(a(b)) = \{a, b\}$$

$$bn(\tau) = \emptyset \;\; bn(\overline{a}(b)) = \{b\} \qquad bn(\overline{a}b) = bn(a(b)) = \emptyset$$

The transition relation $\longmapsto$ of [1] makes uses of labels of the form:

$$\lambda ::= \tau \mid \overline{a}b \mid \overline{a}(b) \mid a(b)$$

and describes behaviours by considering different aspects at once. Labels τ and $a(b)$ are used to describe computations of a process: $P \stackrel{\tau}{\longmapsto} P'$ if P can perform an internal synchronisation and then behaves like P', while $P \stackrel{a(b)}{\longmapsto} P'$ if P behaves like P' after value b is retrieved from channel a. The same relation is used to describe state/spatial configuration of a process: $P \stackrel{\overline{a}b}{\longmapsto} P'$ if $P \equiv P'|\overline{a}b$. Moreover, if the value b is private in P ($P \equiv \nu\, b.P'|\overline{a}b$), P evolves to P' with a label $\overline{a}(b)$ denoting that b is bound in P' and that b is somehow communicated to the environment: b is no more private in P'.

We shall now see how MLTSs can be used to describe behaviour of $A\pi$ processes when computations, spatial configurations and name revelations are modelled separately. We shall first define an MLTS (named $\mathcal{M}_{A\pi}$) that can be used as a semantic model for $A\pi$ processes. The set of states in $\mathcal{M}_{A\pi}$ will be the set of all $A\pi$ processes. Our resources will be the set of terms denoting the availability of messages over channels; indeed π-processes during their computations consume and produce

values over channels.

The computation relation will consider only internal actions and synchronization. For this reason, the set of transition labels in $\mathcal{M}_{A\pi}$ will contain only τ labels.

We let $\mathcal{M}_{A\pi}$ be the MLTS defined as follow:

$$\langle \text{PROC}_{A\pi}, \text{RES}_{A\pi}, \text{LAB}_{A\pi}, \mathbb{N}_{A\pi}, \longrightarrow, \dashrightarrow, \hookrightarrow \rangle$$

where:

- $\text{RES}_{A\pi} = \{\overline{a}b | a, b \in \text{CH}\}$;
- $\text{LAB}_{A\pi} = \{\tau\}$;
- $\mathbb{N}_{A\pi} = \langle \text{CH}, fn, \circ \rangle$;
- $\longrightarrow$, $\dashrightarrow$ and $\hookrightarrow$ are the transition relations induced by the rules of Table 3, plus those induced by the closure of the relations under $\equiv$ as defined in Table 1.

Transition relation describes internal computation of a system caused by internal moves ($\tau.P$) or by process synchronisation. Starting from a process P a resource $\overline{a}b$ can be created leading to process $P|\overline{a}b$. Conversely, a resource $\overline{a}b$ can be consumed in $P \equiv Q|\overline{a}b$ obtaining Q. Finally, a process P reveals a name a, if name a is available over a public channel b. Intuitively, this means that the environment can be able to know the *private name*.

To show that the above is a MLTS, we need to prove that $\mathbb{N}_{A\pi}$ is a named structures for states, resources and transition labels, and that all considered transition relations are preserved by name permutation.

Lemma 3.2 *$\mathbb{N}_{A\pi}$ is a naming structure for $\text{PROC}_{A\pi}$, $\text{RES}_{A\pi}$, $\text{LAB}_{A\pi}$ and CH.*

Proof. The thesis follows easily by noting that:

- for each process P and substitution σ $fn(P)$ is finite, $P \circ \emptyset = P$ and $fn(P \circ \sigma) = \sigma(fn(P))$;
- for each $a \in \text{CH}$, $fn(a) = \{a\}$ and $a \circ \sigma = \sigma(a)$;
- $fn(\tau) = \emptyset$ and $\tau \circ \sigma = \tau$.

□

Lemma 3.3 *$\longrightarrow$, $\dashrightarrow$ and $\hookrightarrow$ are preserved by name permutation.*

Proof. We prove the lemma for $\longrightarrow$. Proofs for $\dashrightarrow$ and $\hookrightarrow$ are similar.

Let σ be a permutation, we have to prove that for each P and Q, if $P \xrightarrow{\tau} Q$ then $P \circ \sigma \xrightarrow{\tau} Q \circ \sigma$.

We proceed by induction on the derivation for $P \xrightarrow{\tau} Q$.

Base of Induction: We can distinguish two cases:

- $P = \tau.Q$ and
- $P = a(b).R|\overline{a}c$ with $Q = R[c/b]$.

In both the cases, for each permutation σ, $P \circ \sigma \xrightarrow{\tau} Q \circ \sigma$.

$$\tau.P \xrightarrow{\tau} P \qquad a(b).P|\overline{a}c \xrightarrow{\tau} P[c/b]$$

$$\frac{P \xrightarrow{\tau} P'}{P|Q \xrightarrow{\tau} P'|Q} \qquad \frac{P \xrightarrow{\tau} P'}{\nu\, a.P \xrightarrow{\tau} \nu\, a.P'}$$

$$P \overset{\oplus\overline{a}b}{\cdots\!\!\rightarrow} P|\overline{a}b \qquad P|\overline{a}b \overset{\ominus\overline{a}b}{\cdots\!\!\rightarrow} P$$

$$\frac{a \neq b}{\nu\, a.(P|\overline{b}a) \hookrightarrow^{a} P|\overline{b}a}$$

Table 3
$\mathcal{M}_{A\pi}$: Transition Relations

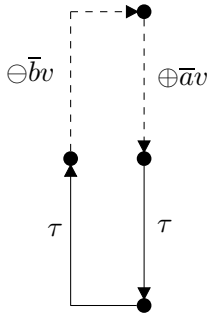

Fig. 1. A portion of the MLTS for PROXY$_1$ of Example 3.1

Induction Hypothesis: For each P and Q, if $P \xrightarrow{\tau} Q$ is derivable in i steps then for each permutation σ, $P \circ \sigma \xrightarrow{\tau} Q \circ \sigma$.

Inductive Step: Let $P \xrightarrow{\tau} Q$ be derivable in $i+1$ steps. We can distinguish three cases:

- $P = P'|R$, $P' \xrightarrow{\tau} Q'$ and $Q = Q'|R$;
- $P \equiv P'$, $P' \xrightarrow{\tau} Q'$ and $Q' \equiv Q$.
- $P = \nu\, a.P'$ $(\sigma(a) = a)$, $P' \xrightarrow{\tau} Q'$ and $Q = \nu\, a.Q'$;

In these cases cases, by using induction hypothesis, for each permutation σ, $P' \circ \sigma \xrightarrow{\tau} Q' \circ \sigma$. Thus the thesis follows easily by considering that for each R and S:

- $(S|R) \circ \sigma = (S \circ \sigma)|(R \circ \sigma)$;
- $\nu\, a.S \circ \sigma = \nu\, a.S \circ \sigma$ $(\sigma(a) = a)$; and
- if $R \equiv S$ then for each σ, $R \circ \sigma \equiv S \circ \sigma$.

□

In Figure 1 you can find a portion of the MLTS describing the behaviour of PROXY$_1$ (see Example 3.1), where dashed arrows denotes *resource* transitions. Please notice only a subset of all the possible transitions are reported in the figure. Indeed, v could be any of the admissible values moreover resources can be added at any state.

Let us now consider the following lemma that permits establishing a correspondence between LTS semantics of [1] and the MLTS semantics presented in this paper.

Lemma 3.4 *The following assertion holds:*

(i) $P \stackrel{\tau}{\longmapsto} P' \Leftrightarrow P \stackrel{\tau}{\longrightarrow} P'$;

(ii) $P \stackrel{\overline{a}b}{\longmapsto} P' \Leftrightarrow P \stackrel{\ominus\overline{a}b}{\cdots\!\!\rightarrow} P'$;

(iii) $P \stackrel{\overline{a}(b)}{\longmapsto} P' \Leftrightarrow \exists P''.P \hookrightarrow^{b} P'' \stackrel{\ominus\overline{a}b}{\cdots\!\!\rightarrow} P'$;

Proof.

(i) $P \stackrel{\tau}{\longmapsto} P'$ if and only if:
- either $P \equiv \nu\,\widetilde{a}.(\tau.Q_1|Q_2)$ and $P' = \nu\,\widetilde{a}.Q_1|Q_2$;
- or $P \equiv \nu\,\widetilde{a}.(\overline{a}b|a(c).Q_1|Q_2)$ and $P' \equiv \nu\,\widetilde{a}.(Q_1[c/b]|Q_2)$.

But the above holds if and only if $P \stackrel{\tau}{\longrightarrow} P'$.

(ii) $P \stackrel{\overline{a}b}{\longmapsto} P'$ if and only if $P \equiv \overline{a}b|P'$. Moreover $P \equiv \overline{a}bP'$ if and only if $P \stackrel{\ominus\overline{a}b}{\cdots\!\!\rightarrow} P'$.

(iii) $P \stackrel{\overline{a}(b)}{\longmapsto} P'$ if and only if $P \equiv \nu\,b.(\overline{a}b|P')$ and $b \neq a$. However, this holds if and only if $P \hookrightarrow^{b} \overline{a}b|P'$ and $\overline{a}b|P' \stackrel{\ominus\overline{a}b}{\cdots\!\!\rightarrow} P'$.

□

4 MLTS Bisimulation and $A\pi$

A methodology for proving properties of process calculi is the one based on behavioural equivalences that requires providing a concrete and an abstract specification of the behaviour of a given system and then establishing that they are "indistinguishable" under appropriate assumptions. Equivalences turn out to be important also because they would permit determining when parts of a system can be *replaced* without changing the behaviour of the whole specification. If we consider the proxy specified in Example 3.1, we could imagine that the τ action, which is performed after a message is retrieved over channel a, models an internal behaviour and the system could be refined with an alternative implementation:

$$\textsc{Proxy}_3 = !\nu\, c.a(x).\overline{c}x|c(y).\overline{b}y$$

Bisimulation has been one of the most successful techniques to establish identifications like the above. We shall now introduce a behavioural equivalence for MLTS that generalizes the bisimulation requirement to the three transition relations of the model. We shall then prove that the induced equivalence coincides with the classical one determined by relation proposed in [1]. The proposed behavioural equivalence is defined over the states of a MLTS. We first define three relations that characterise the bisimulations over the three relations associated to each MLTSs.

Definition 4.1 *Let $\mathcal{M}$ be a MLTS.*

(i) *A relation $\mathscr{R} \subseteq \mathcal{M}^{\mathcal{S}} \times \mathcal{M}^{\mathcal{S}}$ is a resource preserving bisimulation if and only if $\mathscr{R}$ is symmetric and for each $(s_1, s_2) \in \mathscr{R}$ and $r \in \mathcal{M}^{\mathcal{R}}$:*
- *$s_1 \stackrel{\oplus r}{\cdots\!\!\rightarrow} s_1'$ then there exists s_2' such that $s_2 \stackrel{\oplus r}{\cdots\!\!\rightarrow} s_2'$ and $(s_1', s_2') \in \mathscr{R}$;*

- $s_1 \overset{\ominus r}{\cdots\!\!\rightarrow} s_1'$ *then there exists* s_2' *such that* $s_2 \overset{\ominus r}{\cdots\!\!\rightarrow} s_2'$ *and* $(s_1', s_2') \in \mathscr{R}$;

We let $\sim_{\mathcal{R}}$ *denoting the largest resource preserving bisimulation.*

(ii) *A relation* $\mathscr{R} \subseteq \mathcal{M}^{\mathcal{S}} \times \mathcal{M}^{\mathcal{S}}$ *is a revelation bisimulation if and only if* $\mathscr{R}$ *is symmetric and for each* $(s_1, s_2) \in \mathscr{R}$ *and* $n \in \mathcal{M}^{\mathcal{N}}$:

- $s_1 \hookrightarrow^n s_1'$ *then there exists* s_2' *such that* $s_2 \hookrightarrow^n s_2'$ *and* $(s_1', s_2') \in \mathscr{R}$;

We let $\sim_{\mathcal{N}}$ *denoting the largest revelation bisimulation.*

(iii) *A relation* $\mathscr{R} \subseteq \mathcal{M}^{\mathcal{S}} \times \mathcal{M}^{\mathcal{S}}$ *is a* $\mathscr{A}-$*parameterised bisimulation whenever* $\mathscr{A} \subseteq \mathcal{M}^{\mathcal{L}} \times \mathcal{M}^{\mathcal{L}}$, *if and only if* $\mathscr{R}$ *is symmetric and for each* $(s_1, s_2) \in \mathscr{R}$ *and* $\lambda_1 \in \mathcal{M}^{\mathcal{L}}$:

- $s_1 \xrightarrow{\lambda_1} s_1'$ *then there exist* λ_2 *and* s_2' *such that* $s_2 \xrightarrow{\lambda_2} s_2'$, $(\lambda_1, \lambda_2) \in \mathscr{A}$ *and* $(s_1', s_2') \in \mathscr{R}$;

We let $\sim_{\mathcal{L}}^{\mathscr{A}}$ *denoting the largest* $\mathscr{A}-$*parameterised bisimulation.*

Building on the relations above, we can define the wanted bisimulation for MLTS.

Definition 4.2 *Let* $\mathcal{M}$ *be a MLTS,* $\mathscr{A} \subseteq \mathcal{M}^{\mathcal{L}} \times \mathcal{M}^{\mathcal{L}}$, *we let* $\sim^{\mathscr{A}} = \sim_{\mathcal{R}} \cap \sim_{\mathcal{N}} \cap \sim_{\mathcal{L}}^{\mathscr{A}}$

We are now ready to study the relationships between the behavioural equivalence between $A\pi$ processes of [1] and our MLTS equivalence. First of all we have to notice that:

Lemma 4.3

- $\mathcal{M}_{A\pi}$ *is* image-finite *with respect to* $\mathcal{A}_{A\pi}$ *and* $\mathbb{A}_{A\pi}$;
- $\mathscr{A}_{A\pi} = \{(\tau, \tau)\}$ *and* $\mathbb{A}_{A\pi}$ *are* transition respectful *in* $\mathcal{M}_{A\pi}$.

Proof. *Image-finite*ness of $\mathcal{M}_{A\pi}$ with respect to $\mathcal{A}_{A\pi}$ and $\mathbb{A}_{A\pi}$ follows easily from the following:

- for each P and $\overline{a}b$, $\{P' | P \overset{\oplus \overline{a}b}{\cdots\!\!\rightarrow} P'\} = \{P | \overline{a}b\}$;
- for each P and $\overline{a}b$, $\{P' | P \overset{\ominus \overline{a}b}{\cdots\!\!\rightarrow} P'\}$ is $\{Q\}$ if $P \equiv Q \parallel \overline{a}b$ otherwise is the empty set;
- for each P and a, $\{P' | P \hookrightarrow^a P'\} = \{P' | \exists P''.P \equiv \nu\, a.P'$ and $P' \equiv P'' | \overline{a}b\}$ which is finite since in P only a finite number of bound names can occur;
- for each P and α, $\{P' | \exists \lambda.\lambda \in \mathbb{A}[\![\alpha]\!]$ and $P \xrightarrow{\lambda} P'\}$ is equal to $\{P' | P \xrightarrow{\tau} P'\}$ which is finite because in P only a finite number of synchronisations can occur.

Moreover, $\mathscr{A}_{A\pi} = \{(\tau, \tau)\}$. Hence, for each P and λ_1, $\lambda_1 \in \mathbb{A}_{A\pi}[\![\surd]\!]$ while for each $\lambda_2 \in \textsc{Lab}_{A\pi}$, $(\lambda_1, \lambda_2) \in \mathscr{A}_{A\pi}$. Hence, $\mathscr{A}_{A\pi}$ and $\mathbb{A}_{A\pi}$ are transition respectful in $\mathcal{M}_{A\pi}$. □

Moreover, we will show that the equivalence $\sim^{\{(\tau,\tau)\}}$ defined over $A\pi$ processes coincides with the asynchronous bisimulation ($\sim_a$) defined in [1].

Definition 4.4 *A symmetric relation* $\mathscr{R}$ *on* $A\pi$ *terms is a strong* $o\tau-$*bisimulation if* $P \mathscr{R} Q$, $P \overset{\alpha}{\longmapsto} P'$, α *is not an input action, and* $bn(\alpha) \cap fn(Q) = \emptyset$ *implies* $Q \overset{\alpha}{\longmapsto} Q'$ *and* $P' \mathscr{R} Q'$. *Let* $\sim_{o\tau}$ *be the largest* $o\tau-$*bisimulation.*

Definition 4.5 *A relation $\mathscr{R}$ is an asynchronous bisimulation if it is an $o\tau-$bisimulation and whenever $P\mathscr{R}Q$ and $P \stackrel{ab}{\longmapsto} P'$ the following holds:*

- *either $Q \stackrel{ab}{\longmapsto} Q'$ and $P'\mathscr{R}Q'$*
- *or $Q \stackrel{\tau}{\longmapsto} Q'$ and $P'\mathscr{R}(Q'|\overline{a}b)$.*

Definition 4.6 *$\sim_a$ is the largest asynchronous bisimulation.*

Definition 4.7 *Let $\sim_1$ be the largest relation $\mathscr{R}$ such that $\mathscr{R}$ is an $o\tau-$bisimulation and $P\mathscr{R}Q$ implies $(P|\overline{a}b)\mathscr{R}(Q|\overline{a}b)$, for all $\overline{a}b$.*

Theorem 4.8 $\sim_a = \sim_1$

Proof. See [1]. □

Let $P = !\tau.\mathbf{0}$ and $Q = !\tau.\mathbf{0}|a(b).\overline{a}b$, we have that $P \sim_a Q$. Please notice that these processes can be distinguished by using the modal logics defined in [14], [7] and [4].

Theorem 4.9 $\sim^{\{(\tau,\tau)\}} \subseteq \sim_a$

Proof. We first prove that $\sim^{\{(\tau,\tau)\}}$ is an $o\tau-$bisimulation. We have to show that if $P\mathscr{R}Q$, $P \stackrel{\alpha}{\longmapsto} P'$, α is not an input action, and $bn(\alpha) \cap fn(Q) = \emptyset$ implies $Q \stackrel{\alpha}{\longmapsto} Q'$ and $P'\mathscr{R}Q'$.

Thanks to Lemma 3.4, it easy to prove that if $P \sim^{\{(\tau,\tau)\}} Q$ then:

- $P \stackrel{\tau}{\longrightarrow} P' \Rightarrow \exists Q'.Q \stackrel{\tau}{\longrightarrow} Q'$ and $P' \sim^{\{(\tau,\tau)\}} Q'$;
- $P \stackrel{\ominus\overline{a}b}{\cdots\cdots\rightarrow} P' \Rightarrow \exists Q'.Q \stackrel{\ominus\overline{a}b}{\cdots\cdots\rightarrow} Q'$ and $P' \sim^{\{(\tau,\tau)\}} Q'$;;
- $P \hookrightarrow^b P'' \stackrel{\ominus\overline{a}b}{\cdots\cdots\rightarrow} P' \Rightarrow \exists Q', Q''.Q \hookrightarrow^b Q'' \stackrel{\ominus\overline{a}b}{\cdots\cdots\rightarrow} Q'$ and $P'' \sim^{\{(\tau,\tau)\}} Q''$, $P' \sim^{\{(\tau,\tau)\}} Q'$.

We have now to prove that for each $\overline{a}b$ if $P \sim^{\{(\tau,\tau)\}} Q$ then $P|\overline{a}b \sim^{\{(\tau,\tau)\}} Q|\overline{a}b$. For each $\overline{a}b$, $P \stackrel{\oplus\overline{a}b}{\cdots\cdots\rightarrow} P|\overline{a}b$ and $Q \stackrel{\oplus\overline{a}b}{\cdots\cdots\rightarrow} Q|\overline{a}b$. Since $P \sim^{\{(\tau,\tau)\}} Q$, it follows that $P|\overline{a}b \sim^{\{(\tau,\tau)\}} Q|\overline{a}b$ and that $\sim^{\{(\tau,\tau)\}}$ is an asynchronous bisimulation.

Since $\sim_a$ is the largest asynchronous bisimulation, it implies that $\sim^{\{(\tau,\tau)\}} \subseteq \sim_a$.□

Theorem 4.10 $\sim_a \subseteq \sim^{\{(\tau,\tau)\}}$

Proof. We prove that $\sim_a$ is a resource preserving bisimulation, a revelation bisimulation and a $\{(\tau,\tau)\}-$parameterised bisimulation. We have to show that if $P \sim_a Q$ then:

(i) $P \stackrel{\oplus\overline{a}b}{\cdots\cdots\rightarrow} P'$ then there exists Q' such that $Q \stackrel{\oplus\overline{a}b}{\cdots\cdots\rightarrow} Q'$ and $P' \sim_a Q'$.

(ii) $P \stackrel{\ominus\overline{a}b}{\cdots\cdots\rightarrow} P'$ then there exists Q' such that $Q \stackrel{\ominus\overline{a}b}{\cdots\cdots\rightarrow} Q'$ and $P' \sim_a Q'$;

(iii) $P \hookrightarrow^a P'$ then there exists Q' such that $Q \hookrightarrow^a Q'$ and $P' \sim_a Q'$;

(iv) $P \stackrel{\tau}{\longrightarrow} P'$ then there exists Q' such that $Q \stackrel{\tau}{\longrightarrow} Q'$ and $P' \sim_a Q'$.

The above easily follow directly from definition of $\sim_a$ by noting that:

(i) if $P \stackrel{\oplus\overline{a}b}{\cdots\cdots\rightarrow} P'$ then $P' \equiv P|\overline{a}b$;

(ii) if $P \overset{\ominus \overline{a}b}{\cdots\!\!\rightarrow} P'$ then $P \overset{\overline{a}b}{\longmapsto} P'$;

(iii) if $P \hookrightarrow^{b} P'$ then $P' = \overline{a}b|P''$ $(a \neq b)$ and $P \overset{\overline{a}(b)}{\longmapsto} P''$;

(iv) if $P \xrightarrow{\tau} P'$ then $P \overset{\tau}{\longmapsto} P'$.

□

5 Conclusions and future work

In this paper we have proposed a variant of LTS that we called *Multiple Labelled Transition Systems* (MLTS) as a candidate general operational model for distributed calculi with names and mobility. To show usefulness of our proposal we used MLTS to describe the operational semantics of a well known formalism that is the generally recognized as a minimal common denominator of calculi for mobility: the asynchronous π-calculus ($A\pi$) [3,12].

We have also defined a behavioural equivalence between MLTS and proved that, in the case of $A\pi$, this coincides with the asynchronous bisimulation defined in [1].

In a companion paper [9], that can be considered the extended version of the current one, we show versatility of MLTS by using them also for modelling the operational semantics of μKLAIMa calculus for mobile distributed agents with explicit localities [2]. There we also introduce, MoMo, a logic inspired by Hennessy-Milner Logic that consists of a small set of operators to be used to describe properties/behaviours of mobile and distributed systems. Together with the usual logical connectives and the operators for minimal and maximal fixed points, the logics is equipped with operators for describing dynamic behaviours (temporal properties), for modelling resource management (state properties), for keeping into account names handling (nominal properties), and for controlling mobile processes (mobility properties). Temporal formulae, that are interpreted by considering the computation relation describe the actions a system can perform to interact with the environment. These are parameterized with respect to a set A of label predicates. Properties of names, like freshness or revelation, are specified using nominal formulae, whose semantic is given by using relation the revelation and the resource relations.

Our final target is a general framework based on MLTS and MoMo that will sufficient to provide semantics to (and to reason on the behaviour) to terms of process calculi for network aware programming.

Acknowledgement

We would like to thank Diego Latella and Mieke Massink for helpful suggestions and other members of the SENSORIA projects for interesting discussions.

References

[1] Amadio, R. M., I. Castellani and D. Sangiorgi, *On bisimulations for the asynchronous π-calculus*, Theoretical Computer Science **195** (1998), pp. 291–324, an extended abstract appeared in *Proceedings*

of CONCUR '96, LNCS 1119: 147–162.
URL http://www.inria.fr/RRRT/RR-2913.html

[2] Bettini, L., V. Bono, , R. De Nicola, G. Ferrari, D. Gorla, M. Loreti, E. Moggi, R. Pugliese, E. Tuosto and B. Venneri, *The klaim project: Theory and practice*, in: *Global Computing*, Lecture Notes in Computer Science **2874** (2003), pp. 88–150.

[3] Boudol, G., *Asynchrony and the π-calculus (note)*, Rapport de Recherche 1702, INRIA Sophia-Antipolis (1992).
URL http://www.inria.fr/RRRT/RR-1702.html

[4] Caires, L., *Behavioral and spatial observations in a logic for the pi-calculus.*, in: I. Walukiewicz, editor, *FoSSaCS*, Lecture Notes in Computer Science **2987** (2004), pp. 72–89.

[5] Cardelli, L. and A. D. Gordon, *Mobile ambients.*, Theor. Comput. Sci. **240** (2000), pp. 177–213.

[6] Castagna, G. and J. Vitek, *Seal: A framework for secure mobile computations*, in: H. Bal, B. Belkhouche and L. Cardelli, editors, *Internet Programming Languages*, number 1686 in Lecture Notes in Computer Science, Springer, 1999 pp. 47–77.

[7] Dam, M., *Model checking mobile processes*, Journal of Information and Computation **129** (1996), pp. 35–51.

[8] De Nicola, R., G. Ferrari and R. Pugliese, KLAIM*: a Kernel Language for Agents Interaction and Mobility*, IEEE Transactions on Software Engineering **24** (1998), pp. 315–330.

[9] De Nicola, R. and M. Loreti, *Multi labelled transition systems for nominal calculi and their logics*, Matematical Structure in Computer Science (2007), to appear.

[10] Fournet, C., G. Gonthier, J. J. Levy, L. Maranget and D. Remy, *A Calculus of Mobile Agents*, in: U. Montanari and V. Sassone, editors, *Proc. of 7th Int. Conf. on Concurrency Theory (CONCUR'96)*, Lecture Notes in Computer Science **1119** (1996), pp. 406–421.

[11] Hennessy, M. and J. Riely, *Resource access control in systems of mobile agents*, Information and Computation **173** (2002), pp. 82–120.

[12] Honda, K. and M. Tokoro, *An object calculus for asynchronous communication*, in: P. America, editor, *European Conference on Object-Oriented Programming (ECOOP'91)*, LNCS **512** (1991), pp. 133–147.

[13] Milner, R., "Communication and Concurrency," International Series in Computer Science, Prentice Hall, 1989, sU Fisher Research 511/24.

[14] Milner, R., J. Parrow and D. Walker, *Modal logics for mobile processes*, Theoretical Computer Science **114** (1993), pp. 149–171.

[15] Milner, R., J. Parrow and J. Walker, *A Calculus of Mobile Processes, I and II*, Information and Computation **100** (1992), pp. 1–40, 41–77.

[16] Wojciechowski, P. T. and P. Sewell, *Nomadic pict: Language and infrastructure design for mobile agents.*, in: *ASA/MA* (1999), pp. 2–12.