COORDINATED MOBILE AGENTS

Gabriel CIOBANU
Romanian Academy, Institute of Computer Science and A.I.Cuza University of Iași
Blvd. Carol I no.8, 700505 Iași, Romania
Email: gabriel@iit.tuiasi.ro

Coordination of mobile agents is realized by extending a formalism for mobility with timers. The formalism has explicit notions of location and mobility. Timers define timeouts for various resources, making them available only for a certain period of time. We define shortly the syntax of the model and its operational semantics, and use an example to illustrate how the model is working.

1 INTRODUCTION

Coordination is increasingly relevant to complex concurrent and distributed systems, open systems with mobile entities, and dynamically re-configurable evolving systems. These systems accept uncertainty and are difficult to be described. These aspects lead to difficult control schemes where decision-making are distributed widely and unexpectedly. Among the several challenges related to these systems, we can mention the mechanisms for delivery of adequate responses to time-critical demands, coordination methods that for real-time systems, a decentralized control take into account the timed resources.

In this paper we use ambient calculus for describing distributed and mobile agents. In contrast with other formalisms for mobile processes such as the \( \pi \)-calculus [8] whose computational model is based on the notion of communication, the ambient calculus is based on the notion of movement by using an explicit notion of location. An ambient is both a location and the unit of movement. Ambients mobility is controlled by the capabilities \( \text{in}, \text{out}, \text{and} \text{open} \). Inside an ambient there are processes which may exchange messages. Several variants [2, 9, 7] have been proposed by adding and/or removing features of the original calculus [3].

We define the coordinated mobile agents by starting from mobile ambients and defining communication actions, capabilities and ambients as temporal resources. A timer \( \Delta t \) of each temporal resource makes the resource available only for a determined period of time \( t \).

We use the following notions and conventions:

- domain (location) is a space of local computational activity;
- distributed system is an open collection of domains, each capable of hosting a specific execution context (operating systems, rules and resources), and mobile agents which can migrate between domains;
- open systems: a compositional description (process algebra);
- we do not assume a global clock; we use only the local clocks and the relative time of interaction at a specific location.

The aim is to combine the local execution and the migration changing the execution context, using a non-flat (hierarchical) representation of the distributed network. A non-monotonic behaviour is expressed by using timers which trigger a timeout recovery processes when they expire.

We consider an example which can motive and illustrate the model. We assume a student finishing the university activities and moving out of the university building. The student is looking for an available vehicle in order to move to a given location. In front of the university there are a tram stop, a bus stop, and certain cabs. The student has the possibility to use any of the three types of vehicles (tram, bus, and cab) to reach the target location. The scenario used in this paper involves a tram, a bus and two cabs. The coordinated mobile agents try to provide a faithful syntactic description, and a flexible dynamic semantics for such an example. Coordination is more complex when the components are moving. Therefore we define the coordinated agents...
as an extension with timers of the mobile ambients. Timers define timeouts for various resources, making them available only for a determined period of time. Time represents a primary notion; however, since space is dual to time, we think to a model where the notions of location (space) can be explicitly described. Coordination across time is associated with the ability to specify relative starting and ending points of the resource availability. Coordination across space relates to the ability to identify a specific location in which a required resource exists.

2 MOBILE AMBIENTS

Mobile ambients were introduced in 1998 by L. Cardelli and A. Gordon with main purpose to explicitly modelling mobility of agents involved in distributed computation. An ambient is a boundary place where a computation happens (laptop, web-page, virtual address space, etc.). An ambient can be nested within other ambient, and can be moved as whole. An ambient has a name, a collection of local capabilities which control the ambient, and a collection of subambients.

2.1 Formal Syntax

\[
\begin{align*}
\text{n} & \quad \text{names} \\
P,Q & ::= \text{agents} \\
(\nu n) P & ::= \text{restriction} \\
0 & ::= \text{inactivity} \\
P \mid Q & ::= \text{composition} \\
*P & ::= \text{replication} \\
n[P] & ::= \text{ambient} \\
M.P & ::= \text{action} \\
\end{align*}
\]

Entry capability in \( n.P \) instructs the surrounding ambient to enter a sibling ambient \( n \); if \( n \) does not exist, it blocks. The reduction rule of this capability is given by

\[
n[\text{in.n.P} \mid Q] \mid m[R] \rightarrow m[n[P \mid Q]] \mid R
\]

and it is represented graphically as follows

Exit capability out \( n.P \) instructs the surrounding ambient to exit its parent ambient \( n \); if \( n \) does not exist, it blocks. The reduction rule of this capability is given by

\[
m[n[\text{out.n.P} \mid Q] \mid R] \rightarrow n[P \mid Q] \mid m[R]
\]

and it is represented graphically as follows
Coordinated mobile agents

Open capability $\text{open } n.P$ dissolves the ambient $n$ at the same level as the surrounding ambient; if $n$ does not exist, it blocks. The reduction rule of this capability is given by

$$\text{open } n.P \mid n[Q] \rightarrow P \mid Q$$

and it is represented graphically as follows:

The following sequence of pictures describes how the mobile ambients are working:
3 COORDINATED MOBILE AGENTS

We extend the mobile ambients to \((n_{(i,h,d)}^{\nu})[P],Q\), where

- \(n\) denotes an ambient name;
- a timer \(\Delta t\) makes the ambient available for a determined period of time \(t\);
- a capacity \(l\) represents the number of the available resources;
- a weight \(h\) represents the number of resources consumed by ambient \(n\) when it moves into another ambient;
- \(d\) defines a domain where the ambients can use the local computation resources to plan its movements and navigation (local knowledge);
- \(P\) and \(Q\) are processes, where \(Q\) represents a safety process.

According to a local clock, \(\Delta t\) is decreasing its value (it expires when \(t = 0\)); if nothing happens in \(t\) units of time, the ambient \(n_{(i,h,d)}^{\nu}\) is dissolved, process \(P\) is reduced to \(0\), and safety process \(Q\) is executed.

3.1 Syntax

The syntax of the coordinated mobile agents is as follows:

<table>
<thead>
<tr>
<th>n,m</th>
<th>names</th>
<th>P,Q ::= processes</th>
</tr>
</thead>
<tbody>
<tr>
<td>x,y</td>
<td>variables</td>
<td>0</td>
</tr>
<tr>
<td>M ::=</td>
<td>capabilities</td>
<td>(M^{\Delta t}.(P,Q))</td>
</tr>
<tr>
<td>in n n must enter n</td>
<td>((n_{(i,h,d)}^{\nu})[P],Q)</td>
<td>ambient</td>
</tr>
<tr>
<td>out n n must exit n</td>
<td>(P</td>
<td>Q)</td>
</tr>
<tr>
<td>open n n must open n</td>
<td>((vn)P)</td>
<td>restriction</td>
</tr>
<tr>
<td>go (y) (y) migration to (y)</td>
<td>(!{m}^{\Delta t}.(P,Q))</td>
<td>output action</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(?{x}^{\Delta t}.(P,Q))</td>
</tr>
<tr>
<td></td>
<td></td>
<td>*P</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(P + Q)</td>
</tr>
</tbody>
</table>

A migration \(go\ y\) changes the domain \(d\) to a domain \(y\). A capability \(open^{\nu} n .(P,Q)\) evolves to \(P\) whenever in \(\Delta t\) the process becomes sibling to an ambient \(n\); otherwise evolves to \(Q\). An output action \(!\{m\}^{\Delta t}.(P,Q)\) evolves to \(P\) whenever in \(\Delta t\) the process becomes sibling to a process which is intending to capture the name \(m\); otherwise evolves to \(Q\).

The passage of time is described by a discrete time-stepping function \(\phi_{\Delta} : \mathcal{P} \rightarrow \mathcal{P}\).

\[
\phi_{\Delta}(P) = \left\{ \begin{array}{ll}
M^{\Delta(t-1)}.(R,Q) & \text{if } P = M^{\Delta t}.(R,Q), t > 0 \\
Q & \text{if } P = M^{\Delta t}.(R,Q), t = 0 \\
\phi_{\Delta}(R) | \phi_{\Delta}(Q) & \text{if } P = R\{Q\} \\
(vn)\phi_{\Delta}(R) & \text{if } P = (vn)R \\
(n_{(i,h,d)}^{\nu^{(t-1)}})[\phi_{\Delta}(R),Q] & \text{if } P = (n_{(i,h,d)}^{\nu})(R),Q), t > 0 \\
Q & \text{if } P = (n_{(i,h,d)}^{\nu})(R),Q), t = 0 \\
P & \text{if } P = *PQ = 0 \\
\phi_{\Delta}(R) + \phi_{\Delta}(Q) & \text{if } P = R + Q 
\end{array} \right.
\]
\( \phi \) affects the ambients and their capabilities which are not consumed. The consumed capabilities disappear together with their timers.

### 3.2 Semantics

The operational semantics is given by a reduction relation defined by the following reduction rules:

<table>
<thead>
<tr>
<th>Rule</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(R-GTProgress)</td>
<td>( P \rightarrow \varphi_\Delta(P) )</td>
</tr>
<tr>
<td>(R-In)</td>
<td>[ n_{(i',d'),d}^{\varphi'} \left{ \begin{array}{l} m_{(i',d'),d}^{\varphi'}(P,P')</td>
</tr>
<tr>
<td>(R-Out)</td>
<td>[ k_{(i',d'),d}^{d'} \left{ \begin{array}{l} m_{(i',d'),d}^{d'}(P,P')</td>
</tr>
</tbody>
</table>
| (R-Open) | \( h^* \leq l \)
| (R-Go) | \( t^* = 0 \)
| (R-Com) | \!\{m\}^{\varphi'}(P,Q)!\{x\}^{\varphi'}(P',Q') \rightarrow P \rightarrow P'[m/x] \)
| (R-Amb) | \[ n_{(i,d),d}^{\varphi} \left\{ \begin{array}{l} P \rightarrow Q \left( \begin{array}{l} m_{(i,d),d}^{\varphi}(P,Q,R) \left( R \right) \end{array} \right) \end{array} \right] \rightarrow \] |
| (R-LPar) | \[ P \rightarrow Q \left( \begin{array}{l} R \rightarrow P \rightarrow Q \left( R \right) \end{array} \right) \rightarrow \] |
| (R-RPar) | \[ P \rightarrow Q \left( \begin{array}{l} P \rightarrow Q \left( R \right) \rightarrow Q \left( R \right) \end{array} \right) \rightarrow \] |
| (R-Par) | \[ P \rightarrow Q \left( \begin{array}{l} P' \rightarrow Q' \left( \begin{array}{l} P \rightarrow Q' \left( \begin{array}{l} P' \rightarrow Q' \end{array} \right) \end{array} \right) \end{array} \right) \rightarrow \] |
| (R-Res) | \[ P \rightarrow Q \left( \begin{array}{l} (vn)P \rightarrow (vn)Q \left( \begin{array}{l} P \rightarrow Q \left( \begin{array}{l} P' \equiv_p P, Q \equiv_p Q' \left( \begin{array}{l} P \rightarrow Q \left( \begin{array}{l} P' \rightarrow Q' \end{array} \right) \end{array} \right) \end{array} \right) \end{array} \right) \rightarrow \] |

Ambients can interact using these rules only if they are on the same domain. To preserve the timers of \( P \), we define \( \varphi : P \rightarrow P \) by \( \varphi_\Delta(\varphi(P)) = P \).

We denote by \( \rightarrow \) the fact that rules (R-In), (R-Out), (R-Open) and (R-Com) cannot be applied.

If one process evolves by one of (R-In), (R-Out), (R-Open) and (R-Amb), while another one does not perform any reduction, then one of (R-LPar), (R-RPar) should be applied. If more than one process evolve in parallel by applying either (R-In), (R-Out), (R-Open) or (R-Amb), then (R-Par) should be applied. When rules (R-In), (R-Out), (R-Open) and (R-Com) cannot be applied anymore, then (R-GTProgress) is applied.
4 INTERACTION OF COORDINATED MOBILE AGENTS

Various agents are in a distributed system where we point out two domains: univ and campus. In fact, we can imagine a student located at univ and looking to move to campus. It is possible to use either a tram, a bus, or a cab.

\[
\text{student}_{i,1,\text{univ}}[\text{in}^{\text{at}1}\text{tram}.P_1 + \text{in}^{\text{at}2}\text{bus}.P_2 + \text{in}^{\text{at}1}\text{cab}].
\]

\[
\text{at}[\text{out}^{\text{at}4}\text{student}$\langle$\text{campus}$$ \rangle^{\text{at}3}.(P_3, Q_3) \rangle$\ldots
\]

The bus and the tram are moving according to predetermined schedules:

\[
\text{tram}_{i,h,d,\text{init}}[\text{go}^{\text{at}1}s_1, \text{go}^{\text{at}2}s_2 \ldots \text{go}^{\text{at}n}s_n \ldots]
\]

\[
\text{bus}_{i,h,d,\text{init}}[\text{go}^{\text{at}1}p_1, \text{go}^{\text{at}2}p_2 \ldots \text{go}^{\text{at}n}p_n \ldots]
\]

Our scenario involves also two cabs (a hired cab and an available cab).

\[
\text{cab}_{4,1,\text{univ}}[\ast \text{(open}^{\text{at}1}\text{at}.? y^{\text{at}2}.(P, P'), \text{go}^{\text{at}3} y.(Q, Q')) \ldots]
\]

Interaction between the agents student and tram is described by

\[
[\text{student}_{1,1,\text{univ}}[\text{in}^{\text{at}1}\text{tram}]]
\]

\[
[\text{tram}_{i,\text{tram},h,\text{tram},\text{init}}[\text{go}^{\text{at}k+1}s_{k+1} \ldots \text{go}^{\text{at}j}\text{campus} \ldots]]
\]

After \( t_k \) units of time, the tram activates its capability \( \text{go}^{\text{at}k}s_k \) and moves out of the domain univ.

Interaction between the agents student and bus is described by

\[
[\text{student}_{1,1,\text{univ}}[\text{in}^{\text{at}1}\text{bus}]] \text{bus}_{i,\text{bus},h,\text{bus},\text{univ}}[\text{go}^{\text{at}i+1}s_{i+1} \ldots \text{go}^{\text{at}j}\text{campus} \ldots]]
\]

If the bus has free resources, then student can use bus. However the bus have not free resources (its capacity is 0).

Interaction between the agents student and cabs is described by

\[
\text{student}_{1,1,\text{univ}}[\text{in}^{\text{at}1}\text{cab} \ldots]
\]

\[
\text{cab}_{4,1,\text{univ}}[\ast \text{(open}^{\text{at}1}\text{at}.? y^{\text{at}6}.(P, P'), \text{go}^{\text{at}7} y.(Q, Q')) \ldots] \text{cab}_{0,\text{1,univ}}[\ldots]
\]

One cab is not available (its capacity is 0). The other one is available with capacity 4; once hired, its capacity becomes 0. Interaction between the students student and cabs is described by the following evolution steps:

\[
\text{univ}^{\ast}[\text{student}_{1,1,\text{univ}}[\text{in}^{\text{at}1}\text{cab} \ldots]]
\]

\[
\text{cab}_{4,1,\text{univ}}[\ast \text{(open}^{\text{at}1}\text{at}.? y^{\text{at}6}.(P, P'), \text{go}^{\text{at}7} y.(Q, Q')) \ldots] \text{cab}_{0,\text{1,univ}}[\ldots]
\]

\[
\rightarrow \text{univ}^{\ast}[\text{cab}_{4,1,\text{univ}}[\ast \text{(open}^{\text{at}1}\text{at}.? y^{\text{at}6}.(P, P'), \text{go}^{\text{at}7} y.(Q, Q')) \ldots]] \text{campus}^{\ast}[]
\]

\[
\rightarrow \text{univ}^{\ast}[\text{student}_{1,1,\text{univ}}[\text{at}^{\ast}[\text{out}^{\ast1}\text{student}.$\langle$\text{campus}$$ \rangle^{\ast4}.(P_3, Q_3) \rangle$\ldots]]] \text{campus}^{\ast}[]
\]

\[
\rightarrow \text{univ}^{\ast}[\text{student}_{1,1,\text{univ}}[\text{at}^{\ast}[\text{out}^{\ast1}\text{student}.$\langle$\text{campus}$$ \rangle^{\ast4}.(P_3, Q_3) \rangle$\ldots]]] \text{campus}^{\ast}[]
\]

\[
\rightarrow \text{univ}^{\ast}[\text{student}_{1,1,\text{univ}}[\text{at}^{\ast}[\text{out}^{\ast1}\text{student}.$\langle$\text{campus}$$ \rangle^{\ast4}.(P_3, Q_3) \rangle$\ldots]]] \text{campus}^{\ast}[]
\]

\[
\rightarrow \text{univ}^{\ast}[\text{student}_{1,1,\text{univ}}[\text{at}^{\ast}[\text{out}^{\ast1}\text{student}.$\langle$\text{campus}$$ \rangle^{\ast4}.(P_3, Q_3) \rangle$\ldots]]] \text{campus}^{\ast}[]
\]

\[
\rightarrow \text{univ}^{\ast}[\text{student}_{1,1,\text{univ}}[\text{at}^{\ast}[\text{out}^{\ast1}\text{student}.$\langle$\text{campus}$$ \rangle^{\ast4}.(P_3, Q_3) \rangle$\ldots]]] \text{campus}^{\ast}[]
\]

\[
\rightarrow \text{univ}^{\ast}[\text{student}_{1,1,\text{univ}}[\text{at}^{\ast}[\text{out}^{\ast1}\text{student}.$\langle$\text{campus}$$ \rangle^{\ast4}.(P_3, Q_3) \rangle$\ldots]]] \text{campus}^{\ast}[]
\]

\[
\rightarrow \text{univ}^{\ast}[\text{student}_{1,1,\text{univ}}[\text{at}^{\ast}[\text{out}^{\ast1}\text{student}.$\langle$\text{campus}$$ \rangle^{\ast4}.(P_3, Q_3) \rangle$\ldots]]] \text{campus}^{\ast}[]
\]
Coordinated mobile agents

\[ !(\text{campus})^{\Delta_k}(P, Q_3) \mid \text{student}_{(1,1,\text{univ})}[ ] \] campus' [ ]

\[ \rightarrow \text{univ}' [ \text{cach}_{(4,1,\text{univ})}[ \text{go}^{\Delta_k} \text{campus}, (Q, Q') \mid \ast (\text{open}^{\Delta_k} \text{at} \ldots) ] ] \text{cach}_{(4,1,\text{univ})}[ \ast (\text{open}^{\Delta_k} \text{at} \ldots) ] \text{student}_{(1,1,\text{univ})}[ ] \]

5 CONCLUSION

Coordination of mobile agents is given by using explicit notions of location and mobility, as well as some quantitative notions including time and capacity. We extend mobile ambients for controlling mobility in open distributed systems by means of timers, bounded capacities and a simple migration primitive. The evolution of agents is illustrated by an example involving explicit locations, mobility, and non-monotonic behaviour.

The formalism could be used for real application involving mobile agents. A related formalism presented in [1] is used to describe network protocols.

REFERENCES


Received July 21, 2008