Coordination for Mobility: the airport case-study

Antónia Lopes and José Luiz Fiadeiro
University of Lisbon and ATX Software
2nd AGILE meeting, Frauenchiemsee, September 2002

Motivation: The airport case study

We are interested in a design solution that

- involves mobile agents moving from host to host
- distributed the system over hosts at airports and planes
- some of the hosts are themselves mobile
- for flights with stops, the system should also control the
  boarding and deplane of passengers during the intermediary
  stops
- is required to control the check-in and boarding of
  passengers as well as the take-off of planes at airports
- a system that

Used by LMU to illustrate UML extensions to Mobility
The airport case study

• Community with distribution

Motivation

• Architectural design

of the distribution/mobility dimension of a system in its
— lead to primitives that support the explicit representation
— can be extended to mobile systems

coordination based on the separation between computation and
Architectural primitives and modelling primitives

In this talk

•

Provide modelling primitives that support the externalisation of
the mechanisms that are responsible for managing the distribution

Coordination

Computation

Distribution

Architectural Approach to mobility

Development of the structural aspects of an

AGILE

Motivation
The envisaged solution:

- distributes the system over hosts at airports and planes
- includes mobile agents moving from host to host
- some of the hosts are themselves mobile

- logical movement of agents
- physical movement of planes and the movement of hosts they hold

We need a bi-dimensional space in order to model

- includes mobile agents moving from host to host
- distributes the system over hosts at airports and

The envisaged solution:

The airport case study: The Space of Mobility

CommUnity with Distribution: The Space of Mobility

CommUnity does not assume any fixed notion

The space of mobility is constituted by the set of

Loc with a distinguished element \( \perp \rightarrow \text{Loc} \)

Loc

Possible values of a special data type

The application domain in which the system is or will be embedded.

is considered to be adequate for the particular application domain in which the system is or will be embedded.
Component Types

- Passengers
- Planes

Space of Mobility

The airport case study
CommUnity with Distribution: Designs

CommUnity designs are defined in terms of a signature—channels (input, output, private) and action names (shared, private)—location variables (input, output) and local actions.

Design Plane is

The airport case study: Component Types

Plane
design Plane is

outloc l

out fl @l: Flight, prv s @l: [0..3], id@l: PlId, a@l: AirId
do load_lug @l: [s=0 → s:=1]
[] takesoff @l: [s=1 ∧ a ≠ dest(fl) → s:=2 || l:=air.host(l)]
[] lands @l: [s=2 → s:=1 || l:=next(a,f).host(l) || a:=next(a,fl)]
[] unload_lug @l: [s=1 ∧ a=dest(fl) → s:=3]

Position where code is executed

Write frame

State changes

and local actions

— location variables (input, output)

— channels (input, output, private) and action names

CommUnity designs are defined in terms of a signature

Self-inflicted mobility

Plane
The airport case study: Component Types

- Passenger
  - design Passenger
  - inloc l
  - prv s@l: [0..2], seat@l: StId, fl@l: Flight
  - do checkin @l: [s=0 → s:=1]
  - [] boards @l: [s=1 → s:=2]
  - [] leaves @l: [s=2 → s:=1]

Environment-inflicted mobility

Space of Mobility

The airport case study

Coordination Types
- Passengers
- Planes
- Component Types

Coordination of passengers and planes at departure:
- A plane can take off only when all passengers that checked-in are on board.
CommUnity with Distribution: Coordination

The Coordination Model is based on:

- Synchronisation of actions
- Exchanging data through channels
- Bindings of location variables

The mechanisms that establish the coordination of a set of components can be completely externalised from their designs and modelled explicitly as first-class citizens

- Coordination styles that involve the management of the locations of the coordinated parties
- Coordination styles that are location-dependent

System configurations are expressed through diagrams in an appropriate category of designs.

The airport case study: Coordination Types

Coordination of passengers and planes at departure

Appropriate category of designs

- System configurations are expressed through diagrams in an appropriate category of designs

Coordination of passengers and planes at departure

- Coordination styles that involve the management of the locations of the coordinated parties
- Coordination styles that are location-dependent

Design Cable1 is:

```
design Cable1 is
  do a[]:true
  do x:[]
end
```

Design Cable2 is:

```
design Cable2 is
  inloc x
  do a[]:true
  skip
end
```

Design Seq is:

```
design Seq is
  inloc l
  prv s @l: [0..1]
  do ac1 @l: [s=0 → s:=1]
  do ac2 @l: [s=1 → s:=0]
end
```

The mechanisms that establish the coordination of a set of components can be completely externalised from their designs and modelled explicitly as first-class citizens.

- Bindings of location variables
- Exchanging data through channels
- Synchronisation of actions

The Coordination Model is based on:
The airport case study: Coordination Types

Coordination of passengers and planes at departure

\[ \text{design Cable1 is } \text{do } a[\]:true \rightarrow \text{skip} \]
\[ \text{design Cable2 is } \text{inloc } x \text{ do } a[\]:true \rightarrow \text{skip} \]

Passenger Seq Plane

Plane&Seq&Passenger colimit

\[ \text{design Plane&Seq&Passenger is } \text{inloc } l \text{ outloc } l1 \text{ out } fl1@l1: Flight, prv s1@l1: [0..3], id@l1: PlId, a@l1: Airid s2@l2: [0..2], seat@l2: StId, fl2@l2: Flight \text{ s@l1: [0..1]} \]
\[ \text{do load_lug } @l1: [s1=0 \rightarrow s1:=1] \]
\[ \text{takesoff } @l1: [s1=1 \wedge a \neq \text{dest}(fl) \wedge s=1 \rightarrow s1:=2 \mid \mid l1:=\text{air.host}(l1) \mid \mid s:=0] \]
\[ \text{lands } @l1: [s1=2 \rightarrow s1:=1 \mid \mid l1:=\text{next}(a,fl1).host(l1) \mid \mid a:=\text{next}(a,fl1)] \]
\[ \text{unload_lug } @l1: [s1=1 \wedge a=\text{dest}(fl1) \rightarrow s1:=3] \]
\[ \text{checkin } @l2: [s2=0 \rightarrow s2:=1] \]
\[ \text{boards } @l2: [s2=1 \wedge s=0 \rightarrow s2:=2 \mid \mid s:=1] \]
\[ \text{leaves } @l2: [s2=2 \rightarrow s2:=1] \]

This depends on the properties of the media through which component interconnections will be effectively coordinated.

Which distribution pattern should be adopted?

The location of passenger instances was defined to be controlled by the environment.

The management of the location of a passenger is also a form of coordination.

Which distribution pattern should be adopted?

The airport case study: Coordination Types

Coordination of passengers and planes at departure.
Effectiveness of Coordination Mechanisms

The effectiveness of coordination mechanisms put in place through connectors depends on:

- The communication infrastructure available at physical levels.
- The structure of the space of mobility.

Restrictions defined by the communication medium, barriers, etc.

Restrictions due to the existence of tracks, walls, etc.

When $W$ is out of the communication range, the value of $y$ cannot be accessed.

When pos is not reachable from the current position, the movement cannot be carried out.

The effectiveness of coordination mechanisms put in place through connectors depends on:

Effectiveness of Coordination Mechanisms
The semantics of designs is defined in terms of

An algebra $U$ for the data types

An infinite sequence of pairs of binary relations over $U$

$(bti, reach)^i \in \mathbb{N}$

- $n \leq m$ $\iff$ $n$ and $m$ are positions in touch with each other

- $n \, \text{reach} \, m$ $\iff$ position $n$ is reachable from position $m$

- New position has to be reachable from the current one

- An action $g_{l_1, l_2}$ can be executed $\iff$ $[l_1]i \, bti \, [l_2]i$

- An action $g_{l}[D] : [G] \to R$ can be executed $\iff$ $x \, [l]i \, bti \, [x]i$

- $x$ can be read/written from

- New position has to be reachable from the current one

At each step, one of the actions that can be executed is chosen.
The airport case study: a model of the physical world

Only communication between plane and airport hosts is possible. Furthermore this communication is viable iff they are physically co-located.

\[ \mathrm{reach} \iff \mathrm{ph}U(n) = \mathrm{ph}U(m) \land (\mathrm{host}(n) \in \mathrm{U}_{\mathrm{AirId}} \lor \mathrm{host}(m) \in \mathrm{U}_{\mathrm{AirId}}) \]

\[ \lor \mathrm{n=\bot} \lor \mathrm{m=\bot} \]

It is possible to move an agent from an airport host to a plane host.

\[ \mathrm{move} \iff \mathrm{n=\bot} \lor \mathrm{m=\bot} \]

The design of the system is carried out considering the following properties:

- They are physically co-located.
- Coordination between plane and airport hosts when they are physically co-located provides support for the communication infrastructure available at the coordination level.
- The movement of planes obeys to the classical physical rules— teleportation of planes between airports is not possible; they have to fly through the air.
- The design of the system is carried out considering the following properties:
  - They are physically co-located.
  - Coordination between plane and airport hosts when they are physically co-located provides support for the communication infrastructure available at the coordination level.
  - The movement of planes obeys to the classical physical rules— teleportation of planes between airports is not possible; they have to fly through the air.
  - The design of the system is carried out considering the following properties:

The airport case study: Coordination Types

Passengers are mobile agents — boarding triggers their migration to the host of the corresponding plane.

\[ \mathrm{move} \iff \mathrm{host}(n) \in \mathrm{U}_{\mathrm{AirId}} \land \mathrm{host}(m) \in \mathrm{U}_{\mathrm{AirId}} \land \mathrm{ph}(n) = \mathrm{ph}(m) \]

Pattern of distribution and mobility of passenger

The airport case study: a model of the physical world
The airport case study: Coordination Types

Pattern of distribution and mobility of planes

This pattern, initially integrated in the design of the plane, can also be externalised.

The airport case study: Coordination Types

Pattern of distribution and mobility of passengers
In order to investigate if the coordination mechanisms put in place through the connectors can be made effective in the world described previously, we have to provide the initialisation of channels. Then it remains to prove that for every action g of the colimit \[ \text{beg} | \text{Init} = \text{bt, reach loc guard}(g) \]

prove that to provide the initialisation of channels. Then it remains to prove that can be made effective in the world described previously, we have through the connectors.

In order to investigate if the coordination mechanisms put in place
Our approach to system development is based on a methodological separation between — the computations performed by the system components — the mechanisms that coordinated their communication and distribution over the network This separation plays an important role in the way evolution can be conceived:

- The system configuration is subject to dynamic evolution.
- The underlying evolution is a process that needs to be subject to rules.
- New factors, a change in the network environment of hosts in which computing components are running, require at run time the frequent change of coordination mechanisms.

The airport case study: Evolution

- The system configuration is subject to dynamic evolution.
- For instance, the set of instances of Passenger is necessarily dynamic.
- For instance, it is necessary to enforce that all passengers depart.

Change of coordination mechanisms can be conceived:

This separation plays an important role in the way evolution can be conceived:

- Our approach to system development is based on a methodological separation between coordination mechanisms and their communication and the computations performed by the system components.
The airport case study: Evolution

We use Coordination Contexts — a modelling primitive that supports the spec. of the evolution process in terms of:

- Ad hoc services invoked by authorized users
- Programmed reconfigurations (auto-adaptation)

Coordination context:

```
flight(f: flight)
workspace

component types: passenger, plane

connector types: departure, dist_pass

services:
create_passenger(s: SeatId):
  prv pl: plane, pas: passenger
  pas := create passenger with s = 0 and seat = s and fl = f;
  pl := head(match({p: plane | with p.fl = f}));
  create departure(pas, pl) with s = 0;
  create dist_pass(pas, pl) with s = 0;
```

Coordination types:

Component types:

- Passenger
- Plane

Coordination context:

```
create dist_pass(pas, pl) with s = 0;
create departure(pas, pl) with s = 0;
create dist_pass(pas, pl) with s = 0;
```

Workspace services:

- Programmed reconfigurations (auto-adaptation)
- Ad hoc services invoked by authorized users

We use Coordination Contexts — a modelling primitive that supports the spec. of the evolution process in terms of

The airport case study: Evolution
Conclusions

First Results
— We have developed an extension of our architectural approach in which the distribution and mobility aspects of dynamic systems can be modelled explicitly through principles and modelling primitives based on the separation between computation and coordination.

Future Work
— Abstract a general categorical characterisation in order to make this approach available to other formalisms and development platforms.
— Extend the technological separation between computation and coordination for mobile systems.
— Take distribution into account in the process of dynamic reconfiguration of the system.

We have developed an extension of our architectural approach in which the distribution and mobility aspects of dynamic systems can be modelled explicitly through principles and modelling primitives based on the separation between computation and coordination.