Situating A&A ReSpecT for Pervasive Environment Applications

Matteo Casadei     Andrea Omicini

Alma Mater Studiorum – Università di Bologna
{m.casadei, andrea.omicini}@unibo.it

WETICE 2008 - CoMA
Rome, Italy
June 24, 2008
Pervasive Computing: Challenges

When building a pervasive applications, it is key to deal with some critical issues:

- **Context awareness**
  - pervasive systems need to be provided with the ability of perceiving the surrounding (physical) environment and reacting to changes occurring in the environment in a proactive way

- **Context management**
  - perceptions provided by context-aware systems need to be effectively exploited to affect and control specific properties of the environment by enacting actions on the environment itself
Introduction (II)

Requirements

- [Context awareness] → Entities of a pervasive systems need to be situated in the environment
- [Context management] → Entities of a pervasive system need to be coordinated

A suitable paradigm is needed!
Multi-Agent Systems (MAS)

A feasible paradigm for pervasive systems as it promotes:

- decentralization of control
- distribution of intelligence by adopting the abstraction of agent

Some fundamental issues are raised, in particular:

- the semantic conflict between agent *proactiveness* and *reactiveness* needs to be resolved!

  - *reactiveness* in strictly related to the requirement of pervasive components of being situated (*situatedness*)

- distribution of intelligence in MAS requires a suitable model for *agent coordination*
Multi-Agent Systems (MAS)

A feasible paradigm for pervasive systems as it promotes:

- decentralization of control
- distribution of intelligence by adopting the abstraction of agent

Some fundamental issues are raised, in particular:

- the semantic conflict between agent *proactiveness* and *reactiveness* needs to be resolved!
  - *reactiveness* is strictly related to the requirement of pervasive components of being situated (*situatedness*)
- distribution of intelligence in MAS requires a suitable model for *agent coordination*

→ **What is a good meta-model for MAS exploited in pervasive domains?**
The Agents & Artifacts (A&A) Meta-Model

According to this meta-model, MAS are conceived in terms of:

- **Agents**
  - *proactive* entities embodying the intelligence of the systems and oriented toward reaching system’s goal

- **Artifacts**
  - *passive reactive* entities shaping the MAS environment in term of services provided to agent for coordinating their activities, and building up agent environment by mediating between agent and the world outside the MAS

Benefits of A&A

- Separation of concerns between *situatedness* and *proactiveness* makes A&A a suitable model to design MAS for pervasive applications
- We still need to address the issue of coordination among agents by choosing a coordination language for artifact behaviour!
The Agents & Artifacts (A&A) Meta-Model

According to this meta-model, MAS are conceived in terms of:

- **Agents**
  - proactive entities embodying the intelligence of the systems and oriented toward reaching system’s goal

- **Artifacts**
  - passive reactive entities shaping the MAS environment in terms of services provided to agent for coordinating their activities, and building up agent environment by mediating between agent and the world outside the MAS

Benefits of A&A

- Separation of concerns between situatedness and proactiveness makes A&A a suitable model to design MAS for pervasive applications
- We still need to address the issue of coordination among agents by choosing a coordination language for artifact behaviour!
  -> to this end, we chose the A&A ReSpecT coordination language!
Introduction (V)

**A&A ReSpecT**

Coordination language based on *reactions*, embodying the A&A meta-model in order to:

- define artifact behaviour in term of tuple centres (tuple spaces whose behaviour is parametrized by A&A ReSpecT)

We need to extend A&A ReSpecT toward situatedness in order to:

- support the A&A meta-model for pervasive applications

**Goal of This Work**

- Extend A&A ReSpecT toward situatedness
- Show how to exploit A&A based on A&A ReSpecT reactions through a case study regarding the control of temperature in a room
Reactor Specification Tuples in the A&A perspective

- Logic-based language:
  - Tuples are first-order logic terms
  - Prolog unification is the adopted matching mechanism
- Computation within the tuple centre is enabled by defining reactions:
  - A reaction associates an event occurring in a tuple centre with a list of operations to be executed in the tuple centre
- Available primitives are the usual LINDA primitives, which are expressed as Prolog terms
A reaction takes the form $\text{reaction}(E,G,R)$

- $E$ is the template that an event occurring in the tuple centre needs to match for the reaction to be triggered
- $G$ represents a sequence of *guards*, predicates allowing to define conditions over events matching $E$:
  - if all the predicates in $G$ are satisfied, the reaction gets triggered
- $R$ represents a list of reaction goals, namely operations to be executed in the tuple centre when the reaction is triggered
## Situatedness

- The feature of being situated in the environment from both the *spatial* and *temporal* standpoint
  - possibility of perceiving environmental events
- Key feature for a framework to coordinate pervasive applications

## Situatedness in A&A ReSpecT

The original A&A ReSpecT can manage situatedness only from *temporal* viewpoint

> An extension toward spatial situatedness in needed!
Case Study (I): Controlling Environmental Properties of Physical Areas

We focus on a pervasive system aimed at controlling the temperature of a room as a case study to:

- Show the effectiveness of A&A as a meta-model to exploit MAS in pervasive domains
- Introduce A&A ReSpecT extension toward situatedness
- Show how artifact behaviour can be easily enacted by adopting A&A ReSpecT reactions
Case Study (II): A&A-based Architecture

![Diagram of A&A-based Architecture](image)

- **Group of sensors**: \(\text{group of sensors}\)
- **Actuator**: \(\text{actuator}\)
- **Transducer**: \(\text{transducer}\)
Case Study (II): Artifact Structure

Artifacts are internally defined in terms of A&A ReSpecT tuple centres:

- **<<sensor>>** artifacts wrapping real temperature sensors which perceive temperature of different areas of the room
- **<<actuator>>** artifacts wrapping actuators, which act as heating devices so as to control temperature
- **<<aggregator>>** artifact provides an aggregated view of the temperature values perceived by sensors spread in the room since it is linked to **<<sensor>>** artifacts:
  - **<<sensor>>** artifacts update tuples on **<<aggregator>>** artifact through *linkability*
In order to perceive environmental properties, <<sensor>> artifacts need:

- an infrastructural component called transducer that enables the transduction of environmental events in suitable A&A ReSpecT events

On the other hand, A&A ReSpecT addresses situatedness from the language viewpoint.
Case Study (III)

Sensor Artifacts: A&A ReSpecT Behaviour

%(1)
reaction( env_event(temperature, Temp), ( 
    event_time(Time), event_source(sensor(Id)),
    out(sensed_temperature(Id,Temp,Time)),
    tc_aggr@node_aggr ? out(sensed_temperature(Id,Temp)) )
).

%(2)
reaction( out(sensed_temperature(_,Temp,_) ), from_tc, ( 
    in(current_temperature(_)), out(current_temperature(Temp)) )
).

Sensor Artifacts: A&A ReSpecT Behaviour

Events

- In the previous program, reaction (1) is triggered by external events generated by devices in the environment identified by a tuple `env_event(Key, Value)`
  - `Key` and `Value` are Prolog ground terms representing the environmental property under observation and the value of the properties associated with the event
  - in this case `env_event(temperature,Temp)` represents a sensor sending a temperature value `Temp`
Case Study (III)

Sensor Artifacts: A&A ReSpecT Behaviour

Observational Predicates

It is also necessary to observe additional properties of events generated from the environment

- Predicate `event_source(< Value >)`
- For instance `event_source(sensor(Id))`
Aggregator Artifacts: A&A ReSpecT Behaviour

\%(4)
\texttt{reaction( out(sensed\_temperature(Id,Temp)), from\_tc, (}
\begin{align*}
\text{in(total\_temperature(OldTotalTemp),} \\
\text{in(sensed\_temperature(Id,OldTemp)),} \\
\text{TotalTemp is OldTotalTemp - OldTemp + Temp,} \\
\text{out(total\_temperature(TotalTemp),} \\
\text{rd(number\_of\_sensors(SensorNo),} \\
\text{AvgTemp is TotalTemp / SensorNo,} \\
\text{in(average\_temp(_)), out(average\_temp(AvgTemp)) })}
\end{align*}
\text{).}
Case Study (IV): Agents

Agents are goal-oriented and proactive entities that control temperature of the room

1. **get local information from sensor**
   
   `tc_sens@node_i ? rd(current_temperature(Temp_i))`

2. **get global information from aggregator**

   `tc_aggr@node_aggr ? rd(average_temp(AvgTemp))`

3. **deliberate action** determine TempVar based on Temp_i and AvgTemp

4. **act upon actuators** (if TempVar ≠ 0)

   `tc-heat_i@node_i ? out(change_temperature(TempVar))`
Case Study (III)

Actuator Artifacts: A&A ReSpecT Behaviour

%(3)
reaction(
  out(change_temperature(TempVar)),
  from_agent,
  actuator_i ?
    set_env_property(temp_inc,TempVar)
).

Case Study (III)

Actuator Artifacts: A&A ReSpecT Behaviour

Set Environmental Properties

To set properties of devices situated in the environment:

- `<EnvResId> ? set_env_property(<Key>, <Value>)`
- A valid `<EnvResId>` can be `actuator1@137.204.X.X`
Conclusion and Future Work

Conclusion

- Extended A&A ReSpecT to support pervasive computing systems based on MAS and A&A
- Showed a simple case study of a MAS-based pervasive system for controlling environmental properties in physical areas

Future Work

- Concretely realize the abstraction of transducer so as to . . .
- . . . apply A&A ReSpecT to the modelling and design of other pervasive scenarios
Situating A&A ReSpecT for Pervasive Environment Applications

Matteo Casadei    Andrea Omicini

Alma Mater Studiorum – Università di Bologna
{m.casadei,andrea.omicini}@unibo.it

WETICE 2008 - CoMA
Rome, Italy
June 24, 2008