Designing and Programming Organisational Infrastructures for Agents situated in Artifact-based Environments

European PhD

Michele Piunti
michele.piunti@unibo.it

ALMA MATER STUDIORUM
Università di Bologna – DEIS

DEIS - Bologna
January 14th, 2010
Introduction

Tutor  prof. Antonio Natali

Co-Tutor  • prof. Alessandro Ricci
          • prof. Andrea Omicini

European Reviewers  • Mehdi Dastani (Uthrect University - Uthrect, Netherlands)
                    • Juan Antonio Rodriguez Aguilar (Artificial Intelligence Research Institute - Barcellona, Spain)

Collaborations  • Istituto Scienze e Tecnologie della Cognizione - I.S.T.C. - C.N.R., Roma.
                • G2I Groups at Ecole Nationale Supérieure des Mines, St-Etienne - France.
Outline

1. Introduction
2. Programming Organizations
3. Programming Environments
4. Programming Infrastructures
5. Embodying Organisations in MAS Environments
6. Conclusions
Introduction

Complex requirements of nowadays software systems are fostering a paradigm shift in thinking computing:

- **Hardware**
  - Growing computational power and storage
  - Multi-core architectures

- **Software**
  - Programming in the large (distributed and decentralised systems)
  - Fostering either individual either global objectives
  - Coordination, Composition, Cooperation, Scalability, Reliability, Reuse
  - From programming Objects to programming Processes, Actors, Agents

- **Applications**
  - Internet of things (web 2.0, embedded devices, etc.)
  - Cloud Computing, Virtualisation, Electronic Marketplaces, Pervasive and Sociotechnical systems, etc.
  - Scalable semantic integration
  - Authority, Accountability, Security, Trust, Reputation

Users are tightly integrated with IT that enables them to function *organizationally* [Hewitt, 2009]
Organizations, Environments, Multi-Agent Systems

Agent System

“An agent is a computer system that is situated in some environment and that is capable of autonomous action in this environment in order to meet its design objective.” [Wooldridge and Jennings, 1995]

- The notion of environment is intrinsically related to the notion of agent and Multi-Agent Systems (MAS).
- Including both physical and software (“computational”, “virtual”) environments.

Organizations in (Multi) Agent Systems

“Organizations in MAS can be understood as complex entities where a multitude of agents interact, within a structured environment aiming at some global purpose.” [Dignum, 2009]

- An organisation can be seen as a specific infrastructure instrumenting the environment where agents interact.
- Can be ascribed in the context of an interaction space which can be summarized as environment.
Modeling Approaches

Involved entities are established on different concepts and notions, actually:

Agents  mental attitudes, actions, percepts, etc.
Environments  world-of-interest, resources, services, objects, legacy, etc.
Organizations  roles, norms, groups, missions, etc.
Programming Approaches

The trend in organisational MAS is mainly addressed to programming organisations as *middleware* realized by software components which agents can interact by using ad hoc primitives

- AGR/MadKit [Ferber et al., 2003], PowerJade [Baldoni et al., 2008], AMELIE [Esteva et al., 2004], S-MOISE + [Hübner et al., 2005]

Open issues:

- Organizations as “detached” entities of the system;
- Need to “situate” the organisational entity within the workplace where agents are immersed;
- Maintaining the same level of abstraction (*from* design *to* programming models);
- Agent awareness
Situated Organizations

- **MASQ, AGRE** [Stratulat et al., 2009, Báez-Barranco et al., 2006]: integrate different dimensions (agents, environment, interactions, organizations and institutions) into an integral view;

- Distributed normative infrastructures: “normative places” and “normative objects”, reactive entities inspectable by agents and containing readable information about norms [Okuyama et al., 2009].

- Situated Electronic Institutions [Campos et al., 2008]: governor entities allow to bridge environmental structures by instrumenting environments with embodied devices controlled by the institutional apparatus.

- Constitutive rules [Searle, 1997] to bridge the gap between environment and institutional dimensions:
  - The reification of a particular state in a normative place may constitute the realization of a particular institutional fact (e.g., “being on a car driver seat makes an agent to play the role driver”) [Okuyama et al., 2009].
  - “Normative artifact” as a container of *institutional facts* (facts related to the institutional states), and *brute facts* (states related to the concrete workplace where agents dwell) [Dastani et al., 2008]. “Count-as” and “sanctioning” rules allows the infrastructure to recast brute facts to institutional ones and provide normative control.
Challenges

- **Seamless integration of Agents, Environments, Organisations**
  - To provide a unifying approach in programming Agents/Organisations/Environments as a whole;
  - To reconcile agents and their work environments with institutional dimensions (i.e. norms);

- **Grounding Organisations in Agents Work Environment**
  - To integrate in MAS organisational entities which are not modellable as agents

- **Adoption of a real notion of agency**
  - To fully exploit features as mental attitudes (purposes, knowledge), events, perception
  - To reach equilibrium balancing emergent behavior and normative control

- **Interoperability and Openess**
  - To enable agents with/without explicit knowledge of the organisation to participate the organisation;
  - To enable organisations to recognise as players any participating agent and control environment resources
Basic Idea

In complex human organizations, infrastructures are explicitly conceived for easing activities/tasks. Cross disciplinary approach:

- *Intelligent use of Space* [Kirsh, 1995]
- Ergonomics desing and *Cognitive Artifacts* [Norman, 1991]
- Theory of *Social Actions* [Castelfranchi, 1998]

Organization and Environments as a set of functionally related Infrastructures instrumenting agents work environments [Piunti et al., 2009a, Piunti et al., 2009b]

Environments are instrumented with specific Infrastructures
Aiding purposes, easing agent works

Organisational agents, Participant Agents and Infrastructures
Share the same work environment
Structured Approach

Need to define:

- A proper design and programming model, for each involved entity.
- Functional relationships
- Interaction model
Outline

1. Introduction
2. Programming Organizations
3. Programming Environments
4. Programming Infrastructures
5. Embodying Organisations in MAS Environments
6. Conclusions
An organisational specification can be produced using an Organisational Modelling Language (OML) [Boissier et al., 2007]

- Inspired by Human Organizations
- High level of Abstraction

E.g., in MOISE, an organisation is specified by defining the following 3 dimensions [Hübner et al., 2007]¹:

**Structural** Roles, Groups, Relationships

**Functional** Goals, Missions, deadlines (time-to-fulfill)

**Deontic** Norms, Obligations

---

¹ For the adoption of this framework we would thanks the G2I research group at Ecole des Mines of St-Etienne - in particular prof. Olivier Boissier, prof. Jomi Hübner and Rosine Kitio.
**Organisation Modelling**

**Hospital Surgery Room scenario in MOISE:**

(a) Structural Specification in Moise

(b) Functional Specification in Moise

(c) Deontic Specification in Moise

<table>
<thead>
<tr>
<th>id</th>
<th>condition</th>
<th>role</th>
<th>type</th>
<th>mission</th>
<th>TTF</th>
</tr>
</thead>
<tbody>
<tr>
<td>n1</td>
<td></td>
<td>Escort</td>
<td>obligation</td>
<td>mVisit</td>
<td>–</td>
</tr>
<tr>
<td>n2</td>
<td></td>
<td>Patient</td>
<td>obligation</td>
<td>mVisit</td>
<td>–</td>
</tr>
<tr>
<td>n3</td>
<td></td>
<td>Patient</td>
<td>obligation</td>
<td>mPatient</td>
<td>–</td>
</tr>
<tr>
<td>n4</td>
<td></td>
<td>Escort</td>
<td>permission</td>
<td>mPay</td>
<td>5 minutes</td>
</tr>
<tr>
<td>n5</td>
<td>unfulfilled(n4)</td>
<td>Patient</td>
<td>obligation</td>
<td>mPay</td>
<td>5 minutes</td>
</tr>
<tr>
<td>n6</td>
<td></td>
<td>Staff</td>
<td>obligation</td>
<td>mStaff</td>
<td>–</td>
</tr>
<tr>
<td>n7</td>
<td></td>
<td>Doctor</td>
<td>obligation</td>
<td>mDoc</td>
<td>–</td>
</tr>
<tr>
<td>n8</td>
<td>unfulfilled(n5)∧</td>
<td>Staff</td>
<td>obligation</td>
<td>mSan</td>
<td>1 day</td>
</tr>
<tr>
<td></td>
<td>unfulfilled(n4)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n9</td>
<td>fulfilled(n4)∨</td>
<td>Staff</td>
<td>obligation</td>
<td>mRew</td>
<td>1 day</td>
</tr>
<tr>
<td></td>
<td>fulfilled(n5)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n10</td>
<td>unfulfilled(n6)</td>
<td>Doctor</td>
<td>obligation</td>
<td>mStaff</td>
<td>–</td>
</tr>
</tbody>
</table>

Legend:
- min..max
- acquaintance
- communication
- authority
- compatibility
- LINKS
- INTRA-GROUP
- EXTRA-GROUP

mVis: visit
mSan: send
mRew: send fee
mVis: visit
mPay: pay
mDoc: scheme
mStaf: observe
mSan: send
mRew: send
mVis: exit
mVis: visit
mVis: visit
mDoc: scheme
Outline

1 Introduction
2 Programming Organizations
3 Programming Environments
4 Programming Infrastructures
5 Embodying Organisations in MAS Environments
6 Conclusions
A&A meta-model for MAS

- **Agents** Autonomous, goal-oriented, situated, social, reactive and pro-active entities
- **Artifacts** Non-autonomous, automatic/reactive, function-oriented and stateful entities (controllable and observable by agents)
- **Workspaces** Virtual containers of agents and artifacts, defining the topology and the properties of the work environment
Artifact Computational Model

Usage Interface and Observable Properties
Agent-Artifact Interaction

Pragmatic and Epistemic Actions

- Agent-Environment (A-E) interactions are based on the notion of: Usage and Perception
  - No message passing between Agents and Artifacts!
Orthogonality

- Heterogeneous agents (belonging to different agent platforms) work in artifact based environments (belonging to CArtAgO nodes)
  - Integration technologies (bridges) [Ricci et al., 2009].
Outline

1. Introduction
2. Programming Organizations
3. Programming Environments
4. Programming Infrastructures
5. Embodying Organisations in MAS Environments
6. Conclusions
Agents’ work is supported inside the workspace by artifact based infrastructures aiding their activities (missions).
EMI is composed by a set of artifacts needed to situate the organisation inside the workspace, in the context of a specific problem domain.

EMI artifacts are viewed as an ensemble of facilities and resources fostering organisational services (can be linked, governed by workspace rules, etc.)

- Goal Oriented Interaction
- Externalisation
- Coordination

For instance in *Jason*:

```prolog
+!execute_pay
: artifact_id(billing, BmId)
& payment(Params)
<- cartago.use(BmId, pay(Params).
    Receipt).
```
Organisational Artifacts (OA)

OAs are responsible for interpreting a subset of an organisational specification (OML) and managing it at runtime

- ORA4MAS realized as an artifact based infrastructure [Hübner et al., 2009b]
- A&A model (actions and perceptions) define Agent-Organization interactions (A-O)
  - Org. services are artifact operations: adoptRole, leaveRole, commitMission, leaveMission, setGoalAchieved
Organization Management Infrastructure (OMI)

The global functioning of the organisation can be expressed by *institutional facts, norms and rules*

**Normative Organisation Programming Language (NOPL)**

- Organisation is a Normative System
- Organisation specified in OML (i.e., MOISE) can be also translated in a NOPL specification, in terms of norms [Hübner et al., 2009a]

Organisational Management Infrastructure (OMI) is specified using NOPL. OA’s functioning is regulated by embedded NOPL interpreters.
Outline

1. Introduction
2. Programming Organizations
3. Programming Environments
4. Programming Infrastructures
5. Embodying Organisations in MAS Environments
6. Conclusions
Embodied Organisation Rules

Based on *Workspace Rules*, *Emb-Org-Rules* define the functional relationships between Organisation and Environment (O-E):

\[
\text{Emb} - \text{Org} - \text{Rule} ::= \langle \text{count - as} \rangle \mid \langle \text{enact} \rangle \\
\text{count - as} ::= \langle \text{we - ev} \rangle \rightarrow \langle \text{omi - ev} \rangle \\
\text{enact} ::= \langle \text{omi - ev} \rangle \rightarrow \langle \text{emi - ev} \rangle \\
\text{we - ev} ::= \langle \text{omi - ev} \rangle \mid \langle \text{emi - ev} \rangle \\
\text{emi - ev} ::= \langle \text{ws - ev} \rangle \\
\text{omi - ev} ::= \langle \text{ws - ev} \rangle
\]

Constitutive Rules
Count as Rules
Enact Rules
Work Environment Event
EMI Event
OMI Event

Table: Definition of *Emb-Org-Rules* in terms of constitutive rules and events.
The mechanism underlying *Emb-Org-Rules* required a new approach to the A&A computational model:

\[
\begin{align*}
\text{MAS} & = \{ Ws \} \\
\text{Ws} & = \{ \langle ws_n, \langle Ag, Ar, Art, Ev, M, R, t \rangle \rangle \} \\
\text{Ag} & = \{ \langle ag_{id}, ag_s, ag_{Ev}, ag_{pr} \rangle \} \\
\text{Ar} & = \{ \langle ar_{id}, ar_t, I, O, P, V \rangle \} \\
\end{align*}
\]

**Table:** Structures of a Multi-Agent Systems based on Agents&Artifacts model (A&A).

Dynamics described by Operational Semantics
Workspace Events

\[ Ws = \langle Ag, Ar, Art, Ev, M, R, t \rangle \]

Events are generated to signal any relevant change on environments’ state/processes

1. Can be perceived by agents focusing artifacts
2. Can be collected and ranked at the workspace level

\[ ev = \langle ev_t, ev_v \rangle \]

Event pairs (type, value) from Observable Properties:

- \( ev_t = \text{prop\_updated} \): launched since an observable property is updated;
  \( ev_v = \langle ar_{id}, p_n, p_v, t \rangle \)

Events from Operations Execution:

- \( ev_t = \text{op\_req} \): launched since an operation execution is started;
  \( ev_v = \langle ag_{id}, ar_{id}, req_v, t \rangle \)
- \( ev_t = \text{op\_signal} \): event generated during operation execution;
  \( ev_v = \langle ar_{id}, s_t, s_v, t \rangle \)
- \( ev_t = \text{op\_completed} \): launched since an operation execution is terminated;
  \( ev_v = \langle ag_{id}, ar_{id}, op_{req}, t \rangle \)
Workspace Rules

\[ Ws = \langle Ag, Ar, Art, Ev, M, R, t \rangle \]

Need to specify laws governing intra-workspace dynamics and global dynamics inside the workspace.

Workspace as a programmable entity

Event-Condition-Action (ECA) rules: “when ev in the context c apply a”

- \( ev \in Ev \)
- \( c \) refers to observable states \( \in Ar \)
- \( a \) refers to a set of workspace operators

Basic Workspace Operators:

1. \( \text{applyOp}(ar_id, op_{name}, Params) \)
2. \( \text{applyLop}(ar_id, op_{name}, Params) \)
3. \( \text{make}(ar_id, art_n, Params) \)
4. \( \text{dispose}(ar_id) \)
5. \( \text{disable}(ar_id, ag_id, op_{name}) \)
6. \( \text{enable}(ar_id, ag_id, op_{name}) \)
7. \( \text{exclude}(ag_id) \)
8. \( \text{include}(ag_id) \)
Count-As Rules
An event occurring in the system may “count-as” an institutional event and automatically update the organisation

- Once it is situated in a particular institutional context
- Vehicle to address system events to organisational functions
- Promote (automatic) organisational updates

```plaintext
+join_req(Ag) -> make("visitorGroupBoard", "OMI.GroupBoard", [
"moise/hospital.xml","visitGroup"]);
    make("visitorSchBoard", "OMI.SchemeBoard", [
"moise/hospital.xml","visitorSch"]);
    apply("visitorGroupBoard", adoptRole(Ag, "patient"));
    include(Ag).

+op_completed("BillingMachine", Ag, pay) -> apply("visitorSchBoard", setGoalAchieved(Ag, pay_visit)).

+op_completed("Terminal", Ag, sendFee) -> apply("monitorSchBoard", setGoalAchieved(Ag, send_fee)).

+ws_leaved(Ag) -> apply("visitorGroupBoard", leaveRole(Ag, "patient")).
```

Figure: Example of count as rules in the hospital scenario.
Enact Rules

Organisation may produce a control by enacting changes upon the environment (i.e., to promote equilibrium, avoid undesirable states).

- Once there is no need for judgement, **control can be automated** on environment infrastructures
- Even without the intervention of organisational/staff agents

```
+signal("visitorGroupBoard",
    role_cardinality, visitor)
  : true
```

```
+signal("monitorSchBoard",
    goal_non_compliance,
    obligation(Ag,
        ngoa(monitorSch,mRew,send_bill),
        achieved(monitorSch,send_bill,Ag),
        TTF)
  : true
-> exclude(Ag).
```

**Figure:** Example of enact rules in the hospital scenario.
Outline

1. Introduction
2. Programming Organizations
3. Programming Environments
4. Programming Infrastructures
5. Embodying Organisations in MAS Environments
6. Conclusions
Embodied Organisations

**Facts** or status functions, are handled by artifacts’ NOPL interpreters, observable in $Ar_O$

**Functions** provided by artifact operations, exploitable by agents through $Ar_I$

Bidirectional, functional relationships between EMI and OMI established by *Workspace Rules*

- Agents *knows* institutional facts (status functions) observing artifact states
- Agents *pursue* goals exploiting artifact operations (operational functions)
- Organizations *monitor* agents by controlling workspaces entities
- Organizations *regiment* agents by operating over artifacts
Conclusions

Embodiment Rules relate environment and organisational dynamics:

- Global dynamics shaped on workspace events and transparently handled by the system
- No need for agents to bring about (non-native) org. notions
- Agent behavior can be automatically addressed at org. level
- Reglementation and Enforcement over environment resources
Conclusions

Embodiment Rules relate environment and organisational dynamics:

- Global dynamics shaped on workspace events and transparently handled by the system
- No need for agents to bring about (non-native) org. notions
- Agent behavior can be automatically addressed at org. level
- Reglementation and Enforcement over environment resources

Ongoing Works:

- Mechanism of *Workspace Laws* and *Embodied Organisation Rules* can be generalised for defining a wide set of inter-system functional relations (i.e. access control, security);
- Operational Semantics
- 2/3 × implementation - CArtAgO 2.0
Limitations

Aspects we do not address (yet):

- Direct communication between agents (Agent-Agent interaction) through message passing (i.e. ACL) is not under the control of the organisation, actually.

Limitations:

- OMI embodied thanks to Environment Artifacts (need to take their functions into account)
- Dedicated Infrastructures needed for Agent-Agent interaction, actually (i.e. ACL)
- Complex interaction patterns may result in many relationships to be specified between E-O hard to specify


Bibliography III

Perfect Disruption: The Paradigm Shift from Mental Agents to ORGs.
*IEEE Internet Computing*, 13.

Normative programming for organisation management infrastructures.
In *MALLOW Workshop on Coordination, Organization, Institutions and Norms in Agent Systems in Online Communities (COIN-MALLOW 2009)*.

Instrumenting Multi-Agent Organisations with Organisational Artifacts and Agents.
*Journal of Autonomous Agents and Multi-Agent Systems*.

S-moise+: A middleware for developing organised multi-agent systems.


Regimentation and Enforcement

- Regimentation is done by enabling and disabling operation controls (\textit{uic}) on environment artifacts (\textit{visitDoor})
  - This enables or prevents the use of artifacts (CArtAgO implements RBAC)
- Enforcement is done (by staff/organisational agents) by using special artifacts (i.e. the terminal to send fines, the phone to call police, etc.)
Designing and Programming Organisational Infrastructures for Agents situated in Artifact-based Environments

European PhD

Michele Piunti
michele.piunti@unibo.it

ALMA MATER STUDIORUM
Università di Bologna – DEIS

DEIS – Bologna
January 14th, 2010