Multi-Agent Oriented Programming

Introduction to Multi-Agent Oriented Programming with JaCaMo
Outline

Multi-Agent Oriented Programming

Multi-Agent Oriented Programming with JaCaMo

JaCaMo platform
Multi-Agent Oriented Programming (MAOP) aims at programming systems:

- as *organisation* of autonomous *agents* in *interaction* with each other within a shared *environment*,
- by keeping alive from design to execution, concepts pertaining to each of the A/E/I/O families as well as their control/life cycles.

Going beyond each of the A/E/I/O oriented programming approaches
Key features
Multi-Agent Oriented Programming

- **Abstraction**
  - Keeping the concepts alive from design to execution, e.g. no agents sharing and calling OO objects
  - Effective programming models for controllable and observable computational entities

- **Modularity**
  - Away from the monolithic and centralised view

- **Orthogonality**
  - With respect to models, architectures, platforms
  - Support for heterogeneous systems

- **Dynamic extensibility**
  - Dynamic construction, replacement, extension of the entities participating to the system
  - Support for open systems

- **Reusability**
  - Reuse of the entities participating to the system for different kinds of applications
Outline

Multi-Agent Oriented Programming

Multi-Agent Oriented Programming with JaCaMo

JaCaMo meta-model overview

E nvironment dimension

O rganisation dimension

A gent dimension

Integrated dimensions

Synthesis

JaCaMo platform
Outline

Multi-Agent Oriented Programming

Multi-Agent Oriented Programming with JaCaMo
   JaCaMo meta-model overview
      Environment dimension
      Organisation dimension
      Agent dimension
      Integrated dimensions
      Synthesis

JaCaMo platform
JaCaMo meta-model overview

- **ORGANISATION**
  - bakery

- **GROUP**
  - cake

- **WHITEBOARD**
- **ARCHIVE**
- **COM. CHANNEL**
- **BAKERY**
  - workspace
- **TASK SCHEDULER**
- **RESOURCE**
- **CLOCK**

Agents can dynamically join the workspace.
Seamless integrated conceptual dimensions

JaCaMo meta-model overview

Simplified view on JaCaMo meta-model [Boissier et al., 2011]
A seamless integration of three dimensions based on Jason [Bordini et al., 2007],
Cartago [Ricci et al., 2009], Moise [Hübner et al., 2009] meta-models

Forthcoming book in 2020 © MIT Press
Outline

Multi-Agent Oriented Programming

Multi-Agent Oriented Programming with JaCaMo

JaCaMo meta-model overview

Environment dimension

Organisation dimension

Agent dimension

Integrated dimensions

Synthesis

JaCaMo platform
## Environment dimension – Basic concepts

Environment dimension

<table>
<thead>
<tr>
<th>Concept</th>
<th>Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>Environment</td>
<td>Workspace</td>
</tr>
<tr>
<td>Observation</td>
<td>Artifact</td>
</tr>
<tr>
<td>Operation</td>
<td>Environment</td>
</tr>
<tr>
<td>Artifact</td>
<td>Observable property</td>
</tr>
</tbody>
</table>

Simplified conceptual view (A&A meta-model [Omicini et al., 2008])

Simple artifact program:

```java
public class Counter extends Artifact {
    void init(int initialValue) {
        defineObsProperty("count", initialValue);
    }

    @OPERATION void inc() {
        ObsProperty prop = getObsProperty("count");
        prop.updateValue(prop.intValue()+1);
    }
}
```
Environment dimension – Dynamics

Environment dimension

Environment life-cycle
  ▶ Creation/Deletion of Workspaces

Workspace life-cycle:
  ▶ Creation/Deletion of Artifacts
  ▶ Creation/Deletion & Entry/Exit of Agents

Artifact life-cycle:
  ▶ Atomic execution, Success/Failure, Activation/Deactivation of an operation
  ▶ Creation/Deletion/Update of Observable Properties
  ▶ Linking/Unlinking with other artifacts
Outline

Multi-Agent Oriented Programming

Multi-Agent Oriented Programming with JaCaMo

JaCaMo meta-model overview
Environment dimension
Organisation dimension
Agent dimension
Integrated dimensions
Synthesis

JaCaMo platform
Organisation dimension – Basic concepts

Organisation dimension

Simplified Conceptual View (Moise meta-model [Hübner et al., 2009])

Excerpts from organisation program:

Structural spec.

```
<structural-specification>
  <role-definitions>
    <role id="auctioneer"/>
    <role id="participant"/>
  </role-definitions>
  <group-specification id="auctionGroup">
    <roles>
      <role id="auctioneer" min="1" max="1"/>
      <role id="participant" min="0" max="300"/>
    </roles>
  </group-specification>
</structural-specification>
```

Functional spec.

```
<functional-specification>
  <scheme id="doAuction">
    <goal id="auction">
      <argument id="Id"/>
      <argument id="Service"/>
      <plan operator="Sequence">
        <goal id="start"/>
        <goal id="bid" ttf="10 seconds"/>
        <goal id="decide" ttf="1 hour"/>
      </plan>
    </goal>
  </scheme>
</functional-specification>
```

Normative spec.

```
<norm id="n1" type="permission">
  role="auctioneer"
  mission="mAuctioneer"/>
<norm id="n2" type="obligation">
  role="participant"
  mission="mParticipant"/>
```

program in NPL
Organisation dimension – Dynamics

Organisation dimension

Organisation life-cycle
- Creation/Deletion of an Organisation from an Organisation specification
- Entrance/Exit of an agent
- Change of Organisation specification

Organisation structure life-cycle
- Creation/Deletion of a group
- Adoption/Leave of a role

Coordination activity life-cycle
- Creation/End of a schema
- Commitment/Release of a mission
- Change of goal state

Normative Regulation activity life-cycle
- Activation/De-activation of norms
- Fulfillment/Violation of norms
- Enforcement of norms
Outline

Multi-Agent Oriented Programming

Multi-Agent Oriented Programming with JaCaMo

JaCaMo meta-model overview
Environment dimension
Organisation dimension
Agent dimension
Integrated dimensions
Synthesis

JaCaMo platform
Agent dimension – Basic concepts

Agent dimension

Simplified Conceptual View (Jason meta-model [Bordini et al., 2007]):

Simple Agent Program:

```plaintext
happy(bob). // initial belief
!say(hello). // initial goal
/* Plans */
+!say(x) : happy(bob) <- .print(x).
// ...
```

```plaintext
+happy(A) <- !say(hello(A)).
+!say(A) : not today(friday) <- .print(X); !say(X).
+!say(X) : today(friday) <- .print("stop").
-happy(A) : .my_name(A) <- .drop_intention(say(-)).
```

example bob.asl  example carl.asl
Agent dimension – Dynamics

Agent dimension

1. Perceive the environment and update belief base
2. Process new messages
3. Select event
4. Select relevant plans
5. Select applicable plans
6. Create/update intention
7. Select intention to execute
8. Execute one step of the selected intention
Outline

Multi-Agent Oriented Programming

Multi-Agent Oriented Programming with JaCaMo
  JaCaMo meta-model overview
  Environment dimension
  Organisation dimension
  Agent dimension
  Integrated dimensions
  Synthesis

JaCaMo platform
Seamless integrated dimensions

A seamless integration of three dimensions based on Jason [Bordini et al., 2007], Cartago [Ricci et al., 2009], Moise [Hübner et al., 2009] meta-models
Integrating A & A dimensions – Interacting agents

Integrated dimensions – Integrated dimensions

Based on KQML or Jade/FIPA ACL

example alice.asl

example bob.asl
Integrating **A & E** dimensions – Interacting agents

Integrated dimensions – Integrated dimensions

![Diagram of interacting agents](image)

**Based on JaCa bridge**

**Example Alice.asl**

```plaintext
!start.
+count(X) <- .print("counter incremented").
+!start <- .send(bob,tell,happy(bob));
   .send(bob,tell,happy(alice));
   .send(bob,achieve,count).
```

**Example Bob.asl**

```plaintext
happy(bob).
!say(hello).
+!say(X) : happy(bob) <- .print(X).
+!count : count(0) <- inc.
```

example alice.asl  example bob.asl
Integrating **A & O** dimensions

Integrated dimensions – Integrated dimensions

---

Based on ORA4MAS [Hübner et al., 2009]

```
+!contract("SitePreparation", GroupBoardId)
   <- adoptRole(site_prep_contractor);
   focus(GroupBoardId).

+obligation(Ag,Norm,What, Deadline)
   [artifact_id(ArtId)]
   : .my_name(Ag) &
      (satisfied(Scheme,Goal)=What | done(Scheme,Goal,Ag)=What)
   <- !Goal[scheme(Scheme)];
      goalAchieved(Goal)
      [artifact_id(ArtId)].
```
Integrating O & E dimensions

Integrated dimensions – Integrated dimensions

based on Situated Artificial Institution [de Brito et al., 2015]
Outline

Multi-Agent Oriented Programming

Multi-Agent Oriented Programming with JaCaMo
- JaCaMo meta-model overview
- Environment dimension
- Organisation dimension
- Agent dimension
- Integrated dimensions
- Synthesis

JaCaMo platform
Synthesis: MAO Dynamics
Multi-Agent Oriented Programming
The JaCaMo platform
Outline

Multi-Agent Oriented Programming

Multi-Agent Oriented Programming with JaCaMo

JaCaMo platform
  JaCaMo multi-agent platform
  JaCaMo multi-agent system development
Outline

Multi-Agent Oriented Programming

Multi-Agent Oriented Programming with JaCaMo

JaCaMo platform
   JaCaMo multi-agent platform
      JaCaMo multi-agent system development
JaCaMo multi-agent platform

- Multi-agent technologies currently integrated:
  - **Agent** dimension: *Jason* agents [Bordini et al., 2007]
  - **Environment** dimension: CArtAgO platform [Ricci et al., 2009]
  - **Organisation** dimension: *Moise* framework [Hübner et al., 2009]

- Dedicated bridges integrate each of the dimensions altogether:
  - **Agent** – **Environment** integration: c4Jason, c4Jadex [Ricci et al., 2009]
  - **Environment** – **Organisation** integration: count-as/enact rules [Piunti et al., 2009] [de Brito et al., 2015]
  - **Agent** – **Organisation** integration: artifacts dedicated to organisation management [Hübner et al., 2009]

- http://jacamo.sourceforge.net,
  https://github.com/jacamo-lang/jacamo/

Open to integrate other multi-agent technologies
Execution Architecture

JaCaMo multi-agent platform

JaCaMo Node
(CArtAgO, Jason, NOPL Engines + bridges)
Applicative Workspace

Workspace ora4mas

Organisation

Environment
Workspace
Artifact
Observable property
Operation

Agent
Belief
Goal
Action
Plan

Environment
Workspace
Artifact

Organisation
Group
Scheme
Role
Goal

Platform level
Jade, Janus, Java Platforms

mas-grp@ufsc.br

mas-grp@emse.fr
Integration with other technologies

JaCaMo multi-agent platform

► Web 2.0 – http://jaca-web.sourceforge.net
  ► implementing Web 2.0 applications
  ► implementing mobile computing applications on top of the Android platform
  ► building SOA/Web Services applications
► JaCaMo with hypermedia environment – (see next slides)
► Semantic Technologies
  ► JaSA: Semantically Aware Agents
  ► JASDL: Combining agent-oriented programming and semantic web technologies
► JaCaDDM: Distributed Data Mining system founded on the Agents and Artifacts paradigm – https://sourceforge.net/projects/jacaddm/
Outline

Multi-Agent Oriented Programming

Multi-Agent Oriented Programming with JaCaMo

JaCaMo platform
  JaCaMo multi-agent platform
  JaCaMo multi-agent system development
JaCaMo multi-agent system development

▶ Available at:
  ▶ http://jacamo.sourceforge.net/
  ▶ https://github.com/jacamo-lang/jacamo

▶ Documentation:
  ▶ Getting started guides, tutorials, FAQ,
  ▶ Reference documentation on “JaCaMo project files”, on debugging in JaCaMo, on Agent Programming Language, on Organisation Programming Language (most of the JaCaMo documentation is available in the doc folder of the distribution)

All the documentation is available at: http://jacamo.sourceforge.net/

▶ Examples of codes, of demos

▶ Configuration of the platform (.jacamo file in the home directory):
  ▶ to be done after each installation of the platform
Structure of a JaCaMo project:

- **src** groups all the source code of the project
  - **agt** groups all the agents’ code (.asl)
  - **env** groups all the artifacts’ code (.java)
  - **org** groups all the organisations’ code (.xml, .npl)
- one or several JaCaMo project file (.jcm)
- **logging.properties** is the log configuration file

Development environment:

- Use of shell commands:
  - `jacamo-new-project projectName`: new project creation,
  - `jacamo projectName`: project execution,
  - `jacamo-jar fileName`: create a jar with all resources to run the application calling java
- Use of eclipse IDE (JaCaMo plugin for eclipse)
- Use of Gradle
- Use of Docker
Eclipse JaCaMo plugin
JaCaMo multi-agent system development
Agent’s inspector
JaCaMo multi-agent system development

Inspection of agent orgmajordomo

- Beliefs
  - commitment([Italian,mItaly,"jacomoPlan"],[L])
  - commitment([French,mFrench,"jacomoPlan"],[L])
  - commitment([Brazilian1,mBrazil,"jacomoPlan"],[L])
  - commitment([Brazilian2,mBrazil,"jacomoPlan"],[L])
  - current_wsp(cobi_1,"server","427dd8d6-403e-431a-a702-7b11ce574e99")
  - formationStatus(ok)
  - goalState("jacomoPlan", greetings,[French],[French],satisfied)
  - goalState("jacomoPlan", greetings_uk_done,[Italian],[Italian],satisfied)
  - goalState("jacomoPlan", greetings_italy_done,[Italian],[Italian],satisfied)
  - goalState("jacomoPlan", greetings_brazil_done,[Brazilian1,Brazilian2],[Brazilian1,Brazilian2],satisfied)
  - goalState("jacomoPlan", greetings_france_done,[French],[French],satisfied)
  - groups(["jacomoTeam"])
  - my_group("")
  - my_group_id(cobi_2)
  - my_ssh("jacomoPlan")
  - my_ssh_id(cobi_3)

Runs also as an http server
Environment’s inspector
JaCaMo multi-agent system development

Runs also as an http server
Organization structure’s inspector
JaCaMo multi-agent system development

Runs also as an http server
Organization functioning’s inspector
JaCaMo multi-agent system development

Runs also as an http server
Bibliography I


