Multi-Agent Coordination

– Introduction to Multi-Agent Oriented Programming –

O. Boissier

Univ. Clermont Auvergne, IMT Mines Saint-Etienne, LIMOS UMR CNRS 6158, France

CPS2 M2 – Fall 2023

Outline

Current Landscape

Multi-Agent Programming

- Many Agent Oriented Software Engineering (AOSE) **methodology** (Prometheus, Gaia, Tropos, ...) exist!

⇝ Use at least one of these methodologies for analysing and designing your MAS application

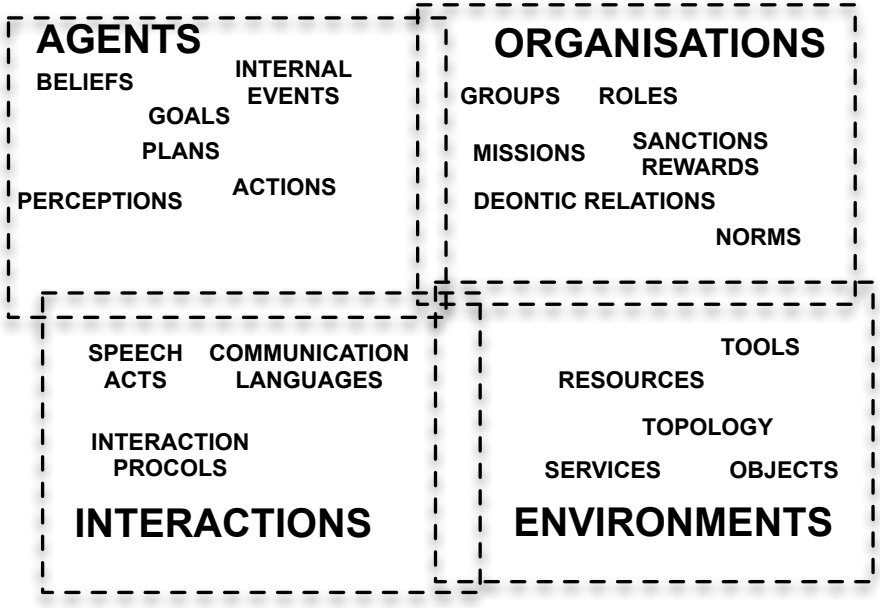
- Many agent languages have efficient and stable interpreters — used extensively in teaching
- All have some programming tools (IDE, tracing of agents' mental attitudes, tracing of messages exchanged, etc.)
- Some are integrating **social** aspects of MAS
- However, there are not yet proper tools to program multi-agent systems!

⇝ some reasons and motivations follow!

Concepts

Multi-Agent Programming

VOWELS' perspective [Demazeau, 1995]:



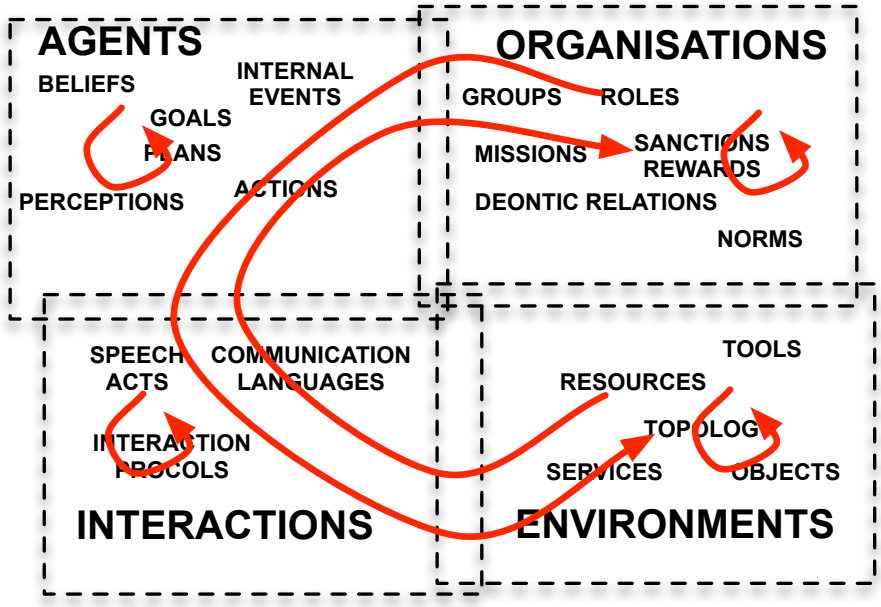
- **O**: abstractions for structuring and ruling the behaviours and interactions of agents in shared environment
- **A**: abstractions for the definition of the decision and reasoning architectures of autonomous entities
- **I**: abstractions for defining interactions among entities
- **E**: abstractions for defining and structuring resource/processing entities shared among the agents

$\rightsquigarrow$ A rich set of abstractions to address applications complexity!

However no consensus on the concepts, on their grouping, on the boundaries

Dynamics

Multi-Agent Programming



- Various life/control cycles among the concepts exist
- Coordination of the system may be programmed using one or several families of concepts [Boissier, 2003]

⇝ Interlacement of the various dynamics into bottom-up / top-down global cycles

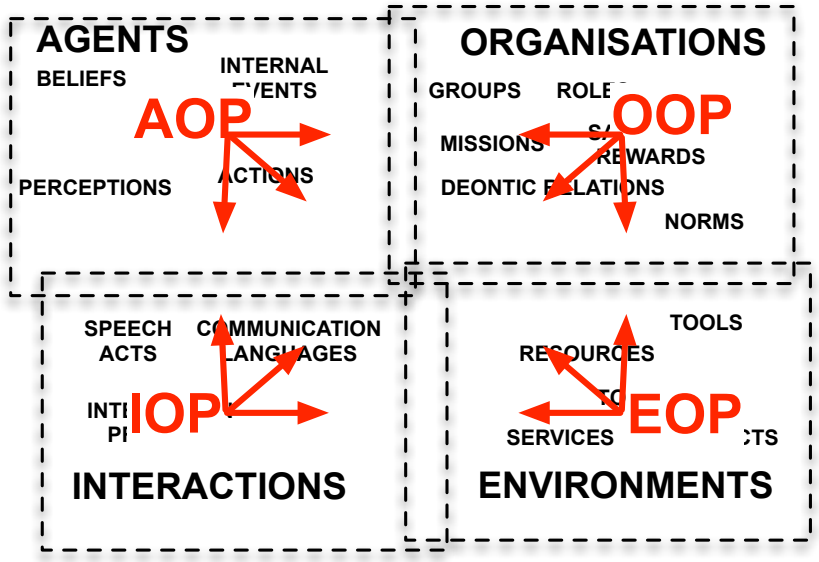
⇝ A rich palette of dynamics to address applications complexity!!

Approaches

Multi-Agent Programming

Agent **O**riented **P**rogramming [Shoham, 1993]

Interaction **O**riented **P**rogramming [Huhns, 2001]



Organisation **O**riented **P**rogramming [Pynadath et al., 1999]

Environment **O**riented **P**rogramming [Ricci et al., 2010]

Platforms

Multi-Agent Programming

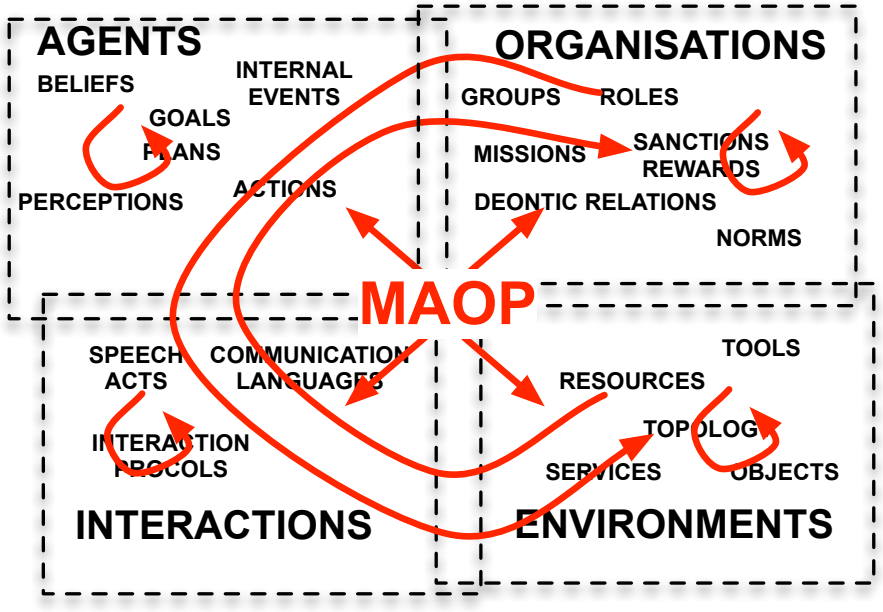
- Platforms with a partial cover of multi-agent abstractions:
  - JADE [Bellifemine et al., 1999, Bellifemine et al., 2000], Java-based Intelligent Agent Componentware [Lützenberger et al., 2013].
  - JACK [Winikoff, 2005, Howden et al., 2001] Jason [Bordini et al., 2007] 2APL [Dastani et al., 2003], GOAL [Hindriks, 2009] Jadex [Pokahr et al., 2005, Pokahr et al., 2014]
- Integrated programming approaches:
  - Volcano platform [Ricordel and Demazeau, 2002], MASK platform [Occello et al., 2004], MASQ [Stratulat et al., 2009], Situated E-Institutions [Campos et al., 2009], MANET [Tampitsikas et al., 2011], ANTE [Cardoso et al., 2016], Electronic Institutions - EI/EIDE [Noriega and de Jonge, 2016], InstAL [Padget et al., 2016], ROMAS/MAGENTIX2 [García et al., 2016], RTEC [Artikis et al., 2016], SARL [Rodriguez et al., 2014] BRAHMS[Sierhuis et al., 2003]
  - $\rightsquigarrow$ Socio-cognitive systems [Aldewereld et al., 2016]

However some families of concepts lose their control & visibility!

Difficulty to integrate and keep alive the families of concepts!

Outline

Multi-Agent Oriented Programming



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.

$\rightsquigarrow$ Going beyond each of the A/E/I/O oriented programming approaches

Key features

Multi-Agent Oriented Programming

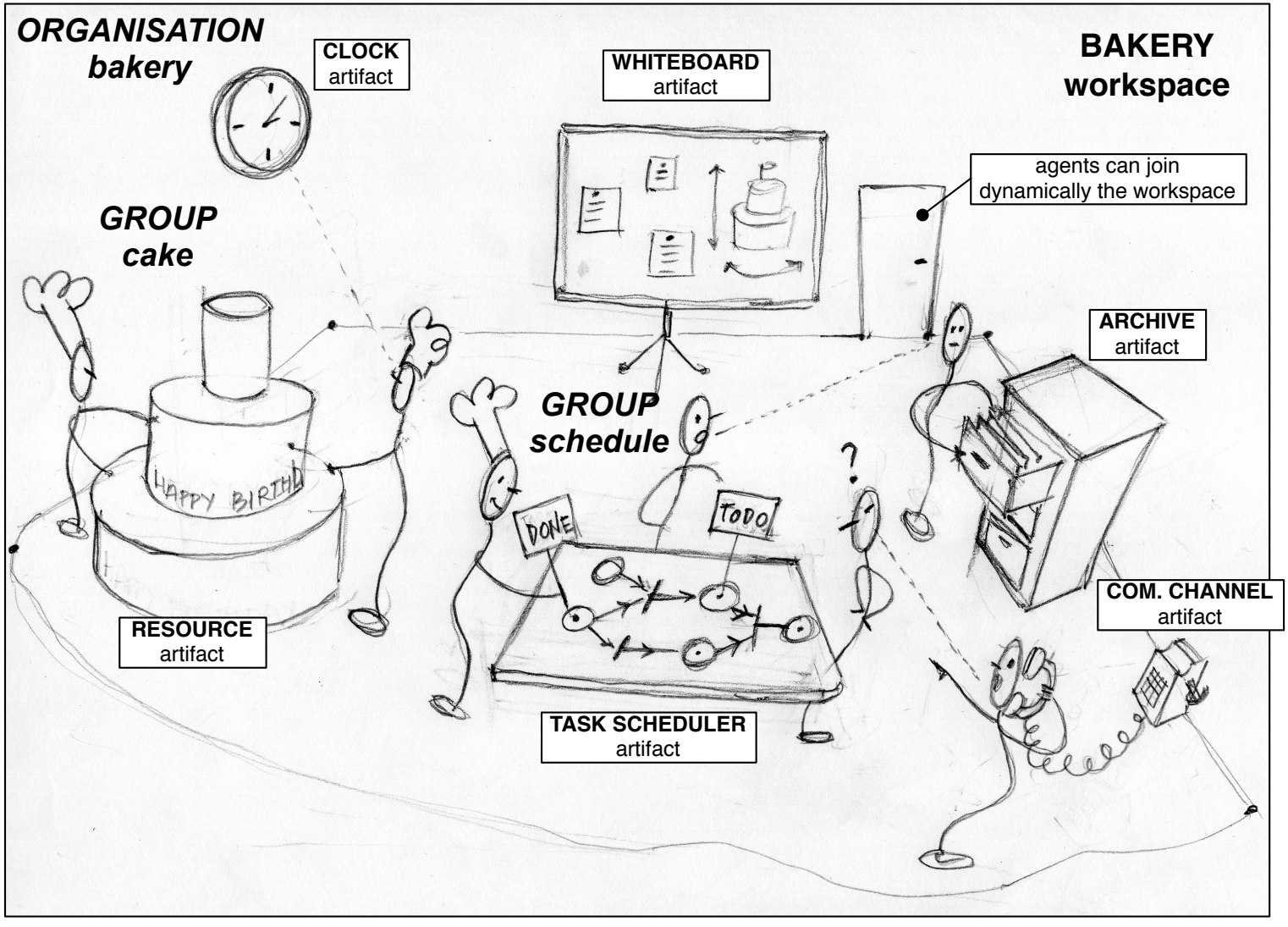
- **Abstraction**
  - keeping the concepts alive from design to execution, e.g. no agents sharing and calling OO objects
  - effective programming models for controllable and observable computational entities
- **Modularity**
  - away from the monolithic and centralised view
- **Orthogonality**
  - wrt 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

Outline

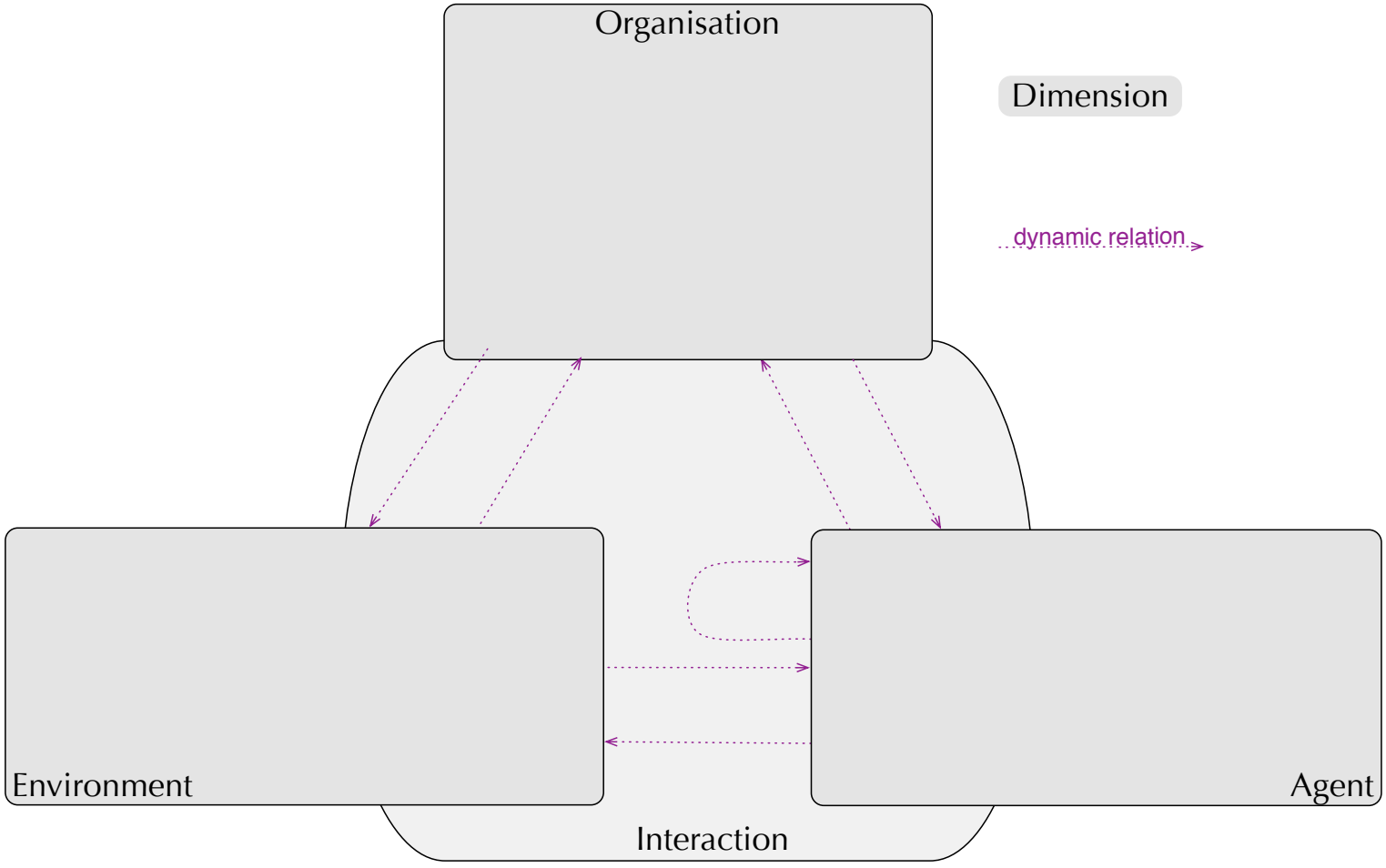
JaCaMo vision

JaCaMo meta-model overview



Seamless integrated conceptual dimensions

JaCaMo meta-model overview



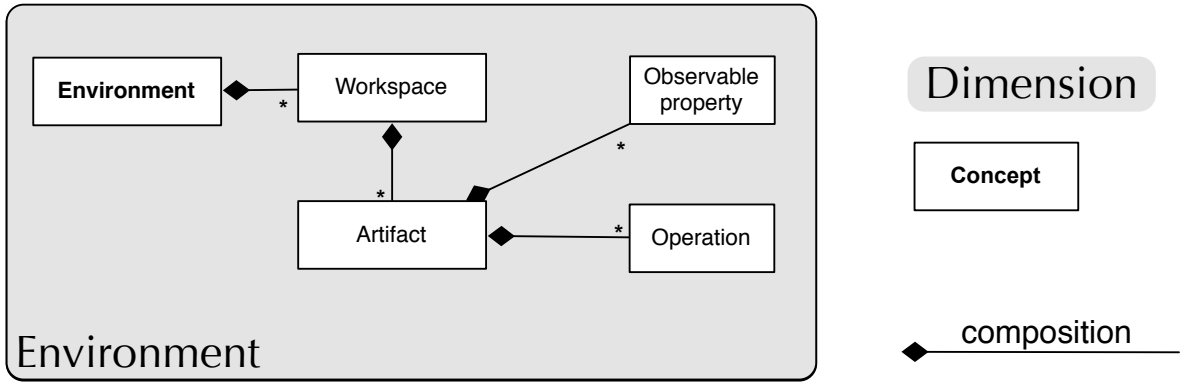
Simplified view on JaCaMo meta-model [Boissier et al., 2020, Boissier et al., 2011]

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

Outline

Environment dimension – Basic concepts

Environment dimension



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

Simple artifact program:

```
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 life-cycle

- Creation/Deletion of Workspaces

Workspace life-cycle:

- Creation/Deletion of Artifacts
- Creation/Deletion & Entry/Exit of Agents

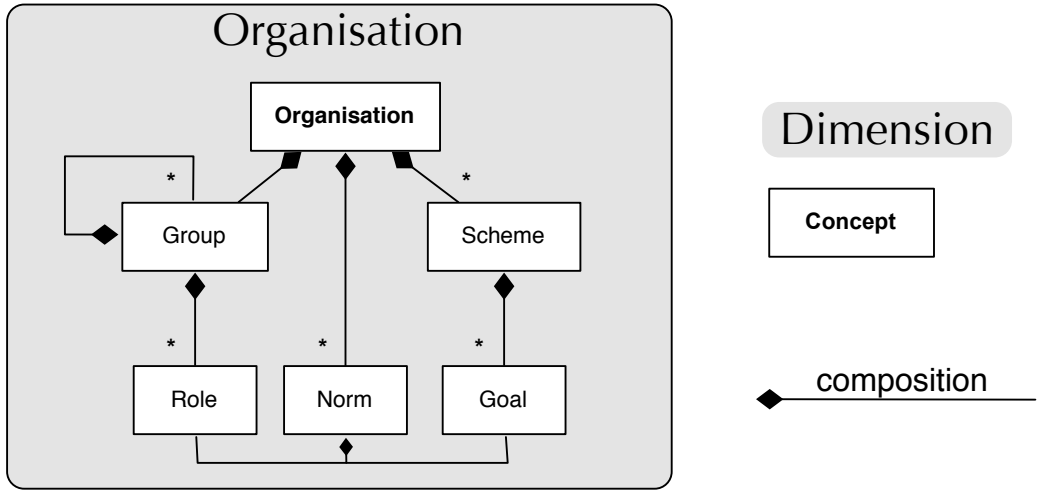
Artifact life-cycle:

- Atomic execution, Success/Failure, Activation/Deactivation of an operation
- Creation/Deletion/Update of Observable Properties
- Linking/Unlinking with other artifacts

Outline

Organisation dimension – Basic concepts

Organisation dimension



Simplified Conceptual View ($\mathcal{M}$oise meta-model [Hübner et al., 2009])

Excerpts from organisation program:

```
<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>
```

Structural 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>
  <mission id="mAuctioneer" min="1" max="1">
    <goal id="start" />
    <goal id="decide" />
  </mission>
```

Functional spec.

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

Normative spec.

```
norm n1 : plays(A, auctionneer, G) ->
  forbidden(A,n1,plays(A,participant,G),
    'forever').
```

program in NPL

Organisation dimension – Dynamics

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

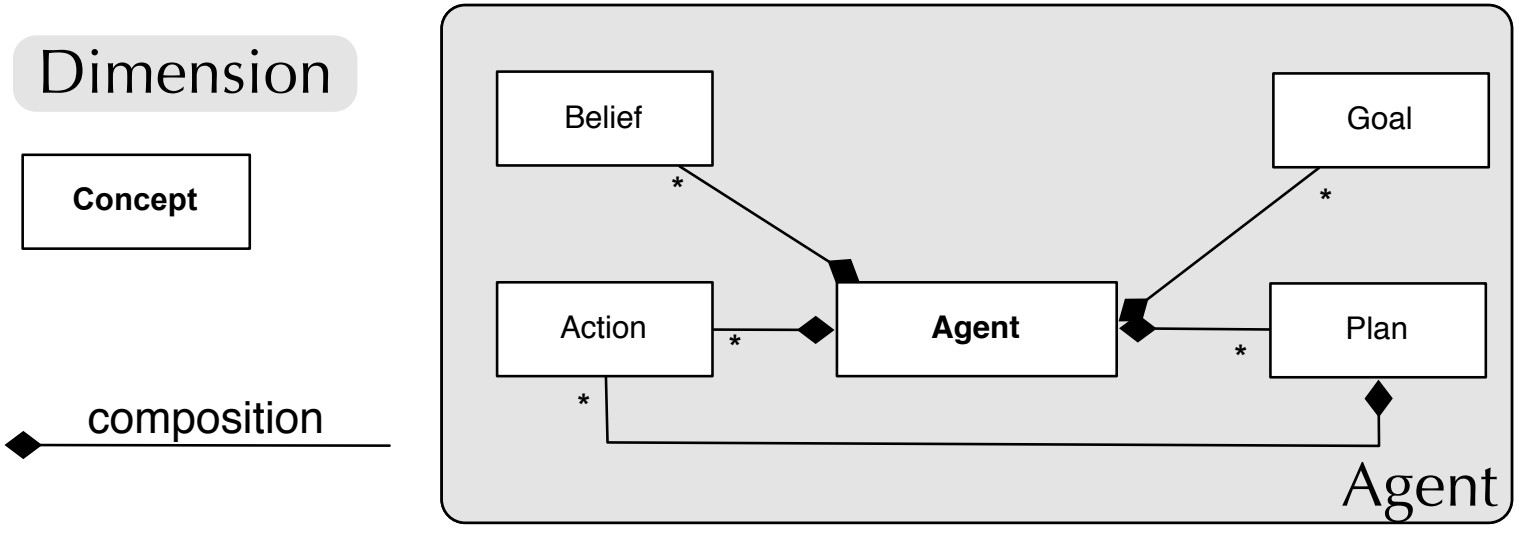
Normative Regulation activity life-cycle

- Activation/De-activation of norms
- Fulfillment/Violation of norms
- Enforcement of norms

Outline

Agent dimension – Basic concepts

Agent dimension



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

Simple Agent Program:

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

example bob.asl

```
+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 carl.asl

Agent dimension – Dynamics

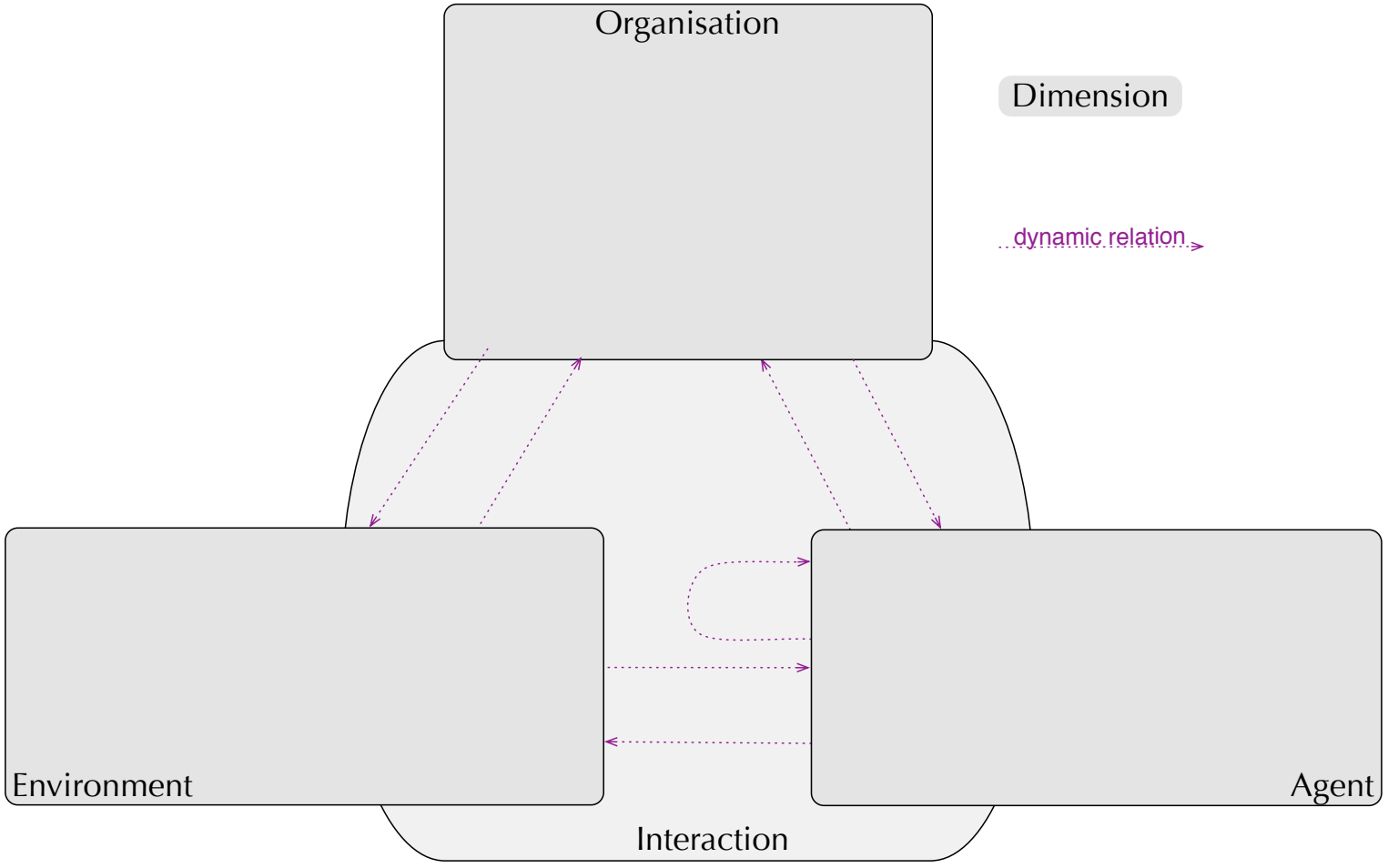
Agent dimension

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

Seamless Integrated Dimensions

Integrated Dimensions

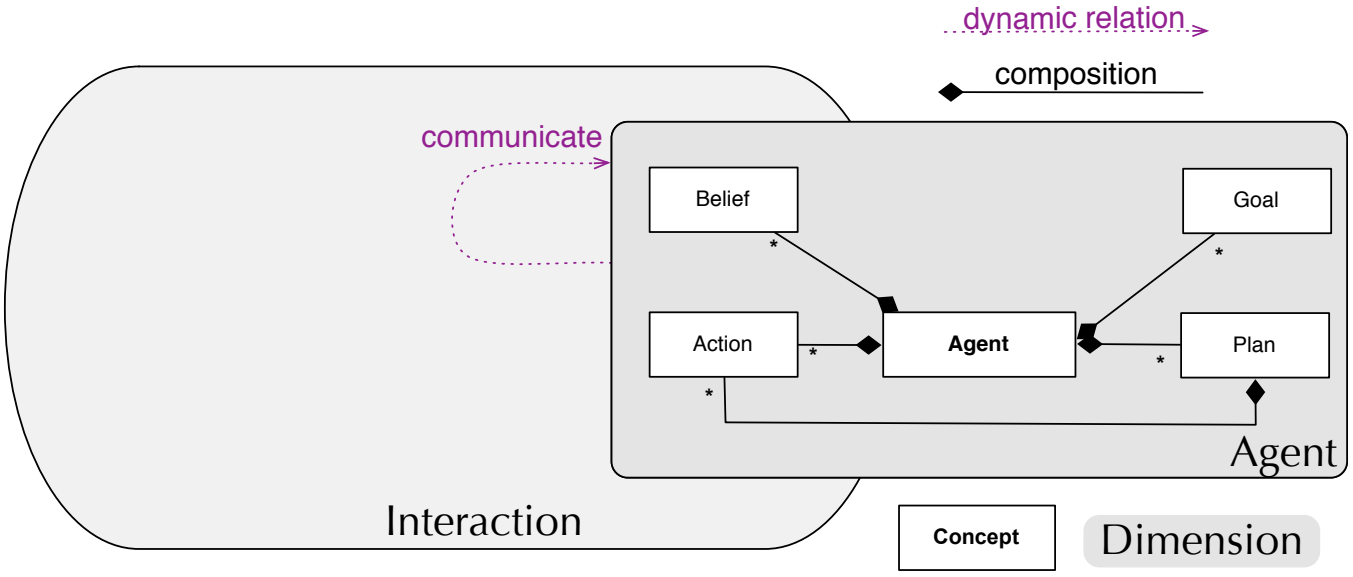


Simplified view on JaCaMo meta-model [Boissier et al., 2020, Boissier et al., 2011]

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

Integrating **A** & **A** dimensions – Interacting agents

Integrated Dimensions



based on KQML or Jade/FIPA ACL

```
!start.
+!start <- .send(bob,tell,happy(bob));
           .send(bob,tell,happy(alice));
           .send(bob,achieve,count).
```

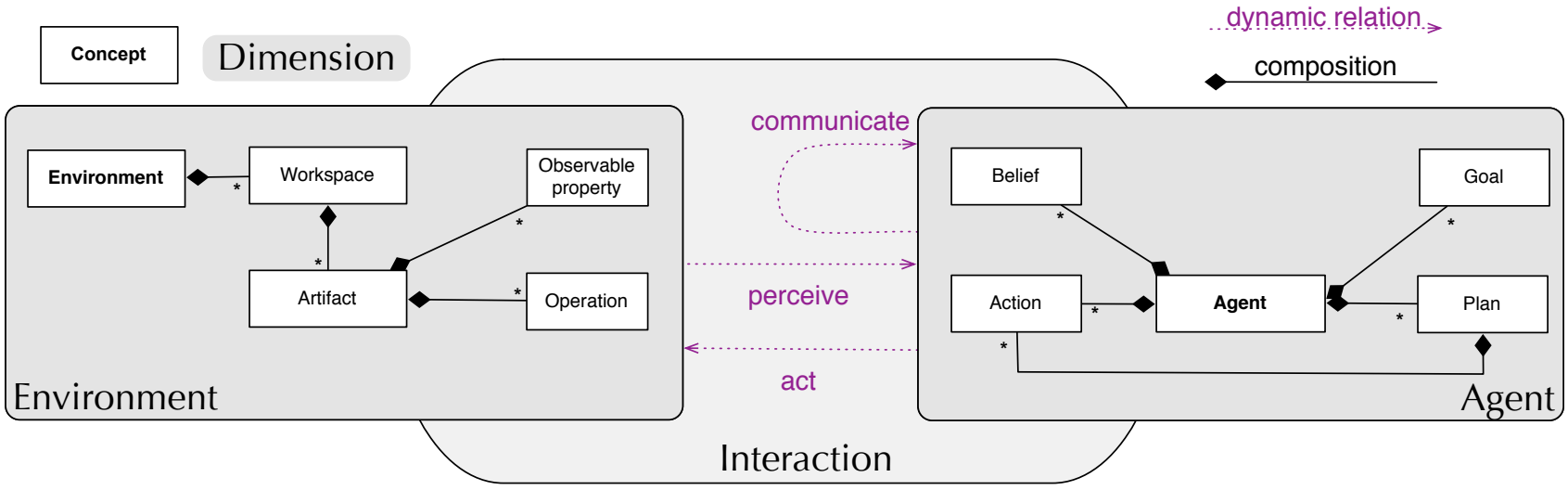
example alice.asl

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

example bob.asl

Integrating **A** & **E** dimensions – Interacting agents

Integrated Dimensions



based on JaCa bridge

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

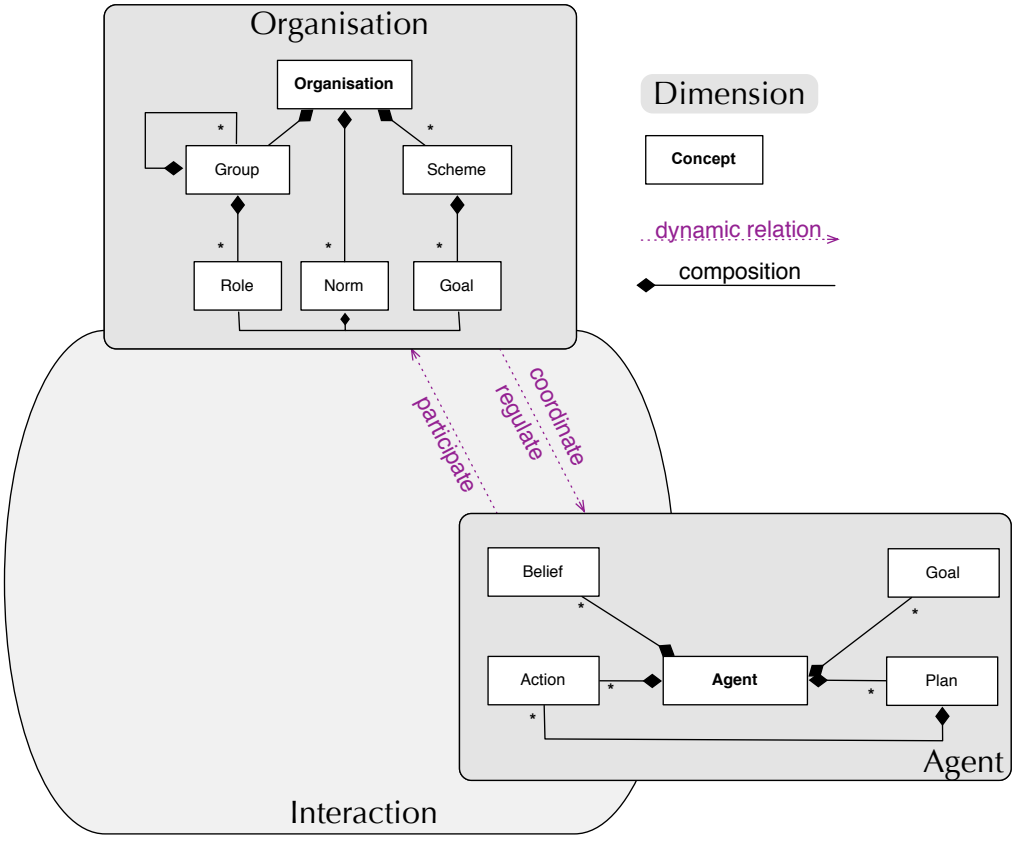
example alice.asl

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

example bob.asl

Integrating A & O dimensions

Integrated Dimensions

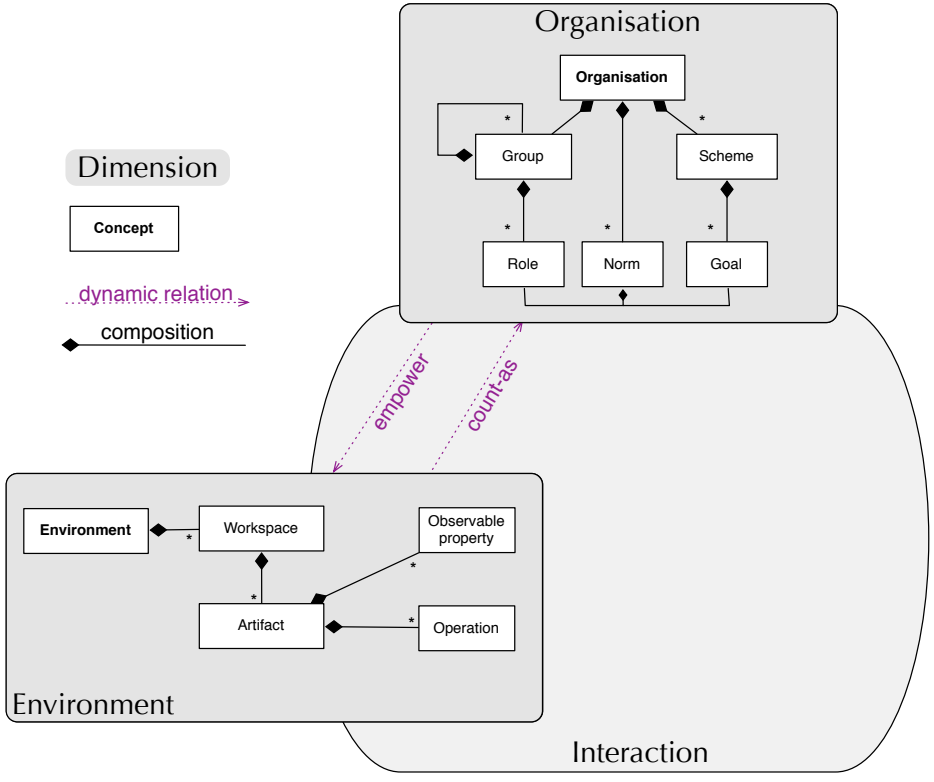



```
+!contract("SitePreparation",
       GroupBoardId)
   <- adoptRole(site_prep_contractor);
      focus(GroupBoardId).

+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)].
```

based on ORA4MAS [Hübner et al., 2009]

Integrating O & E dimensions

Integrated Dimensions



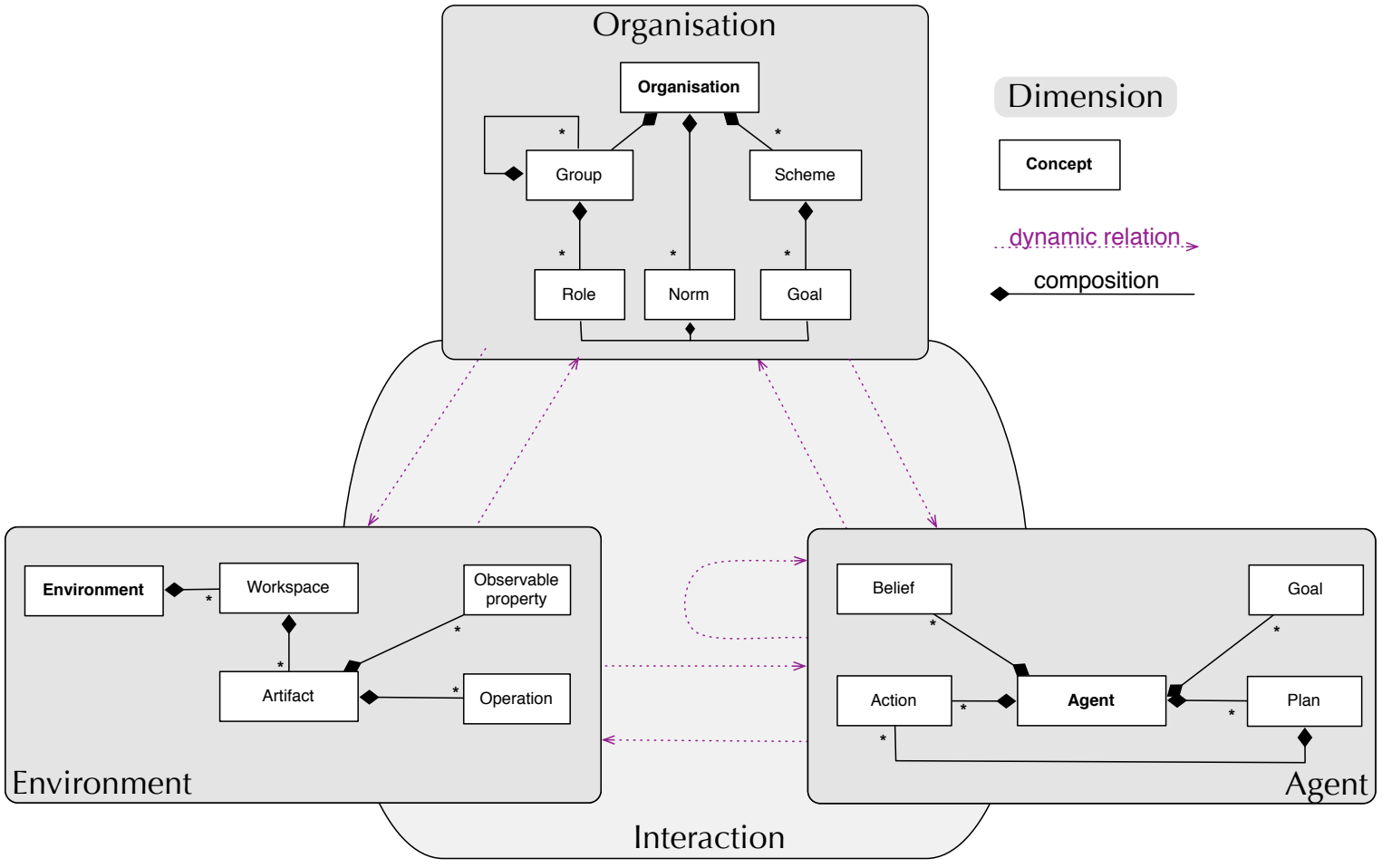

```
institution_id : bhInst.
status_functions:
states: play(A,R,G),
  responsible(G,S),
  committed(A,Mission,S),
  achieved(S,G,A),
  done(S,G,A).

constitutive_rules:
...
2:
currentWinner(auction_for_SitePreparation,Agent)
count-as play(Agent,site_prep_contractor,
              "hsh_group")
while nticks(clock,Time)&(Time>=8000).
...
12:
play(A,house_owner,"hsh_group")
count-as committed(A,management_of_house_building,
                   "bhsch")
while responsible("hsh_group","bhsch").
...
22:
count-as achieved("bhsch",site_prepared,Agent)
when prepareSite[sai__agent(Agent)].
...
```

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

Outline

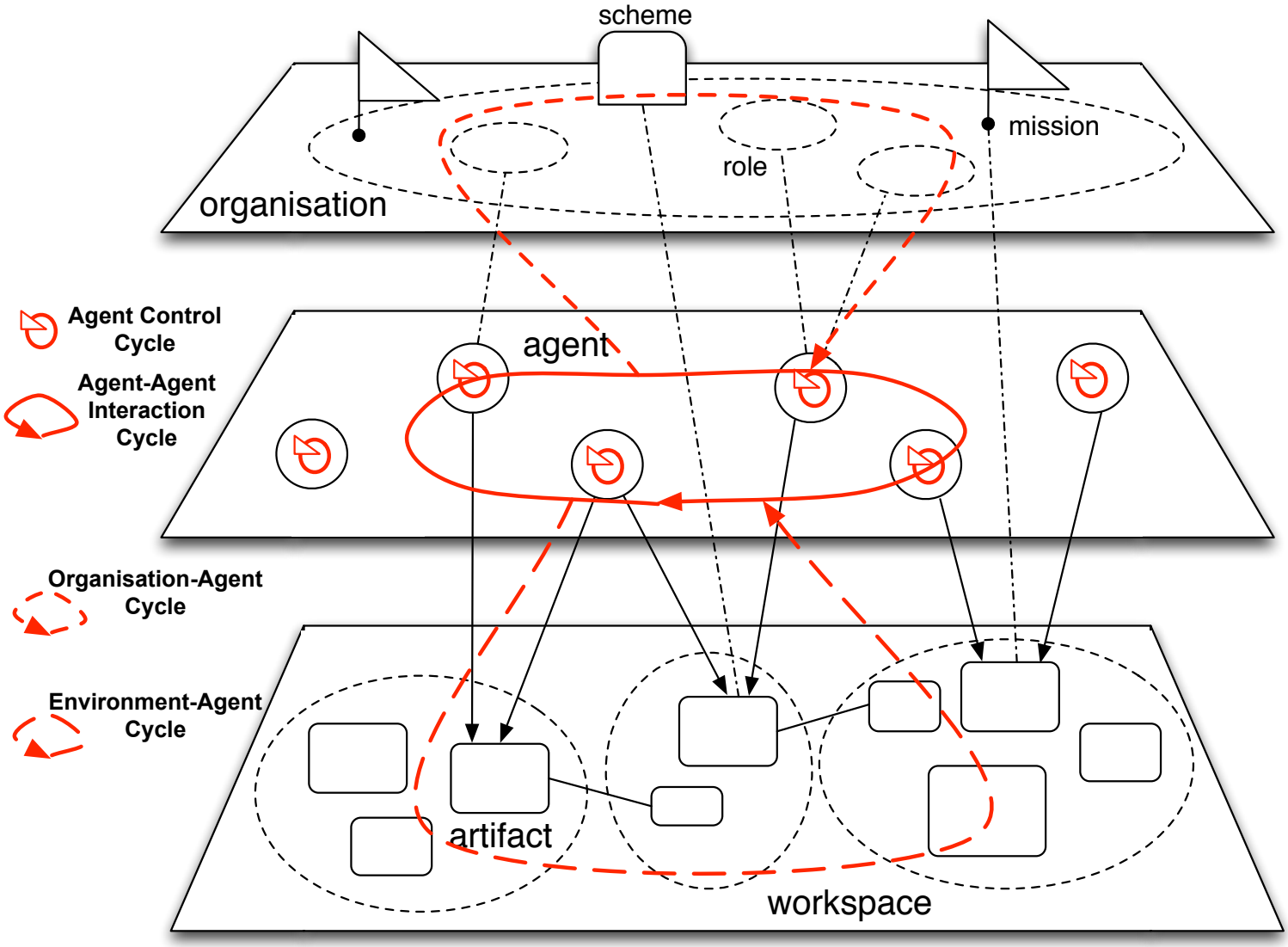
Multi-Agent Oriented Abstractions

Synthesis



Multi-Agent Oriented Dynamics

Synthesis



Multi-Agent Oriented Programming

The JaCaMo platform

Outline

Outline

JaCaMo multi-agent platform

- Multi-agent technologies currently integrated:
  - **A**gent dimension: *Jason* agents [Bordini et al., 2007]
  - **E**nvironment dimension: CArtAgO platform [Ricci et al., 2009]
  - **O**rganisation dimension: $\mathcal{M}$oise framework [Hübner et al., 2009]

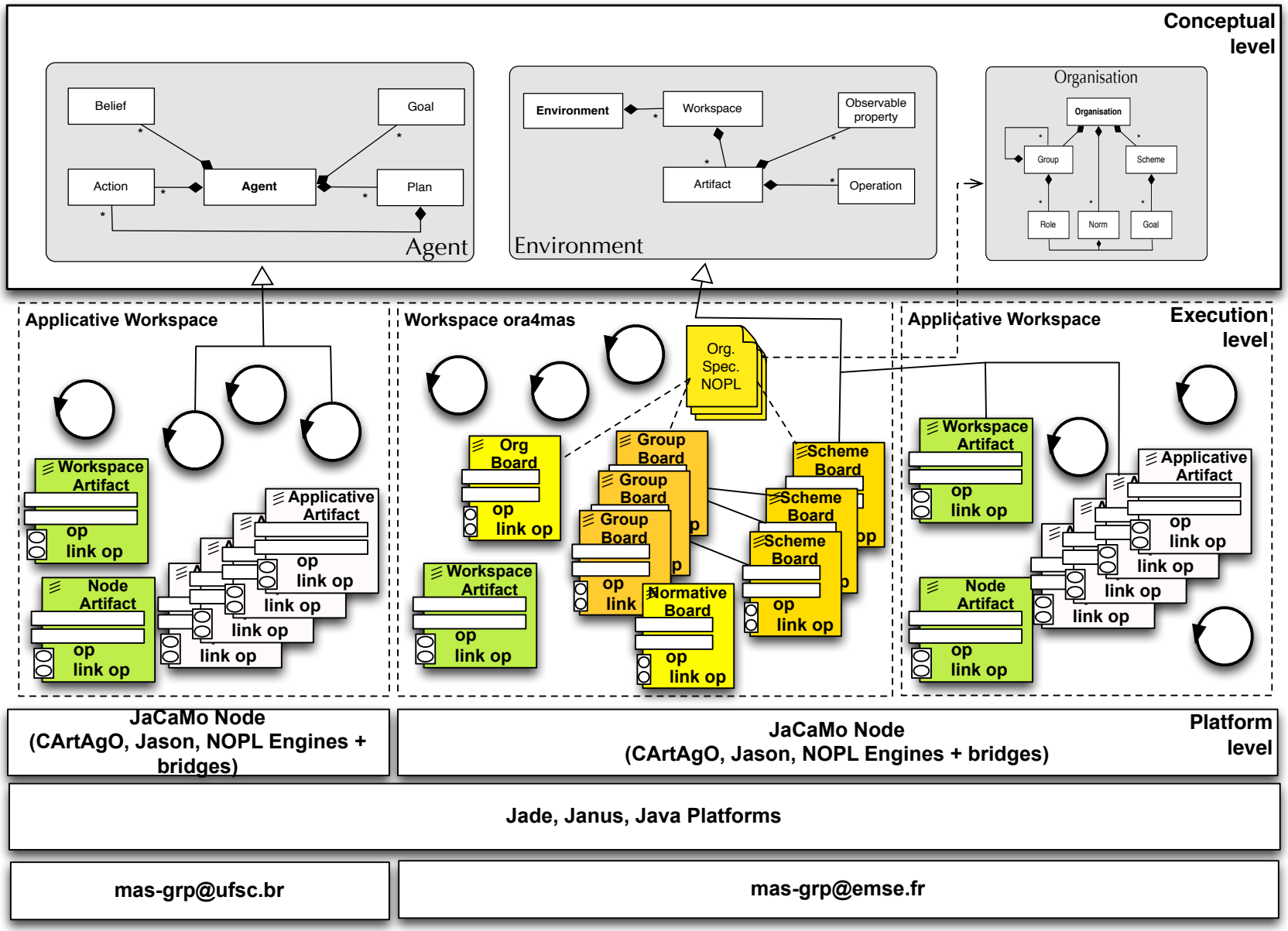
- Dedicated bridges integrate each of the dimensions altogether:
  - **A**gent – **E**nvironment integration: c4Jason, c4Jadex [Ricci et al., 2009]
  - **E**nvironment – **O**rganisation integration: count-as/enact rules [Piunti et al., 2009] [de Brito et al., 2015]
  - **A**gent – **O**rganisation integration: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

$\rightsquigarrow$ `http://jacamo.sourceforge.net`, `https://github.com/jacamo-lang/jacamo/`

Open to integrate other multi-agent technologies

Execution Architecture

JaCaMo multi-agent platform



Integration with other technologies

JaCaMo multi-agent platform

- Web 2.0 – `http://jaca-web.sourceforge.net`
  - implementing Web 2.0 applications
- Android Platforms – `http://jaca-android.sourceforge.net`
  - implementing mobile computing applications on top of the Android platform
- Web Services – `http://cartagows.sourceforge.net`
  - building SOA/Web Services applications
- Arduino Platforms – `http://jacamo.sourceforge.net`
- JaCaMo with hypermedia environment – (see next slides)
- Jason-ROS: modular interface between Jason, CArtAgO, and ROS – `https://github.com/lsa-pucrs/jason-ros-releases/releases`
- 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

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/`

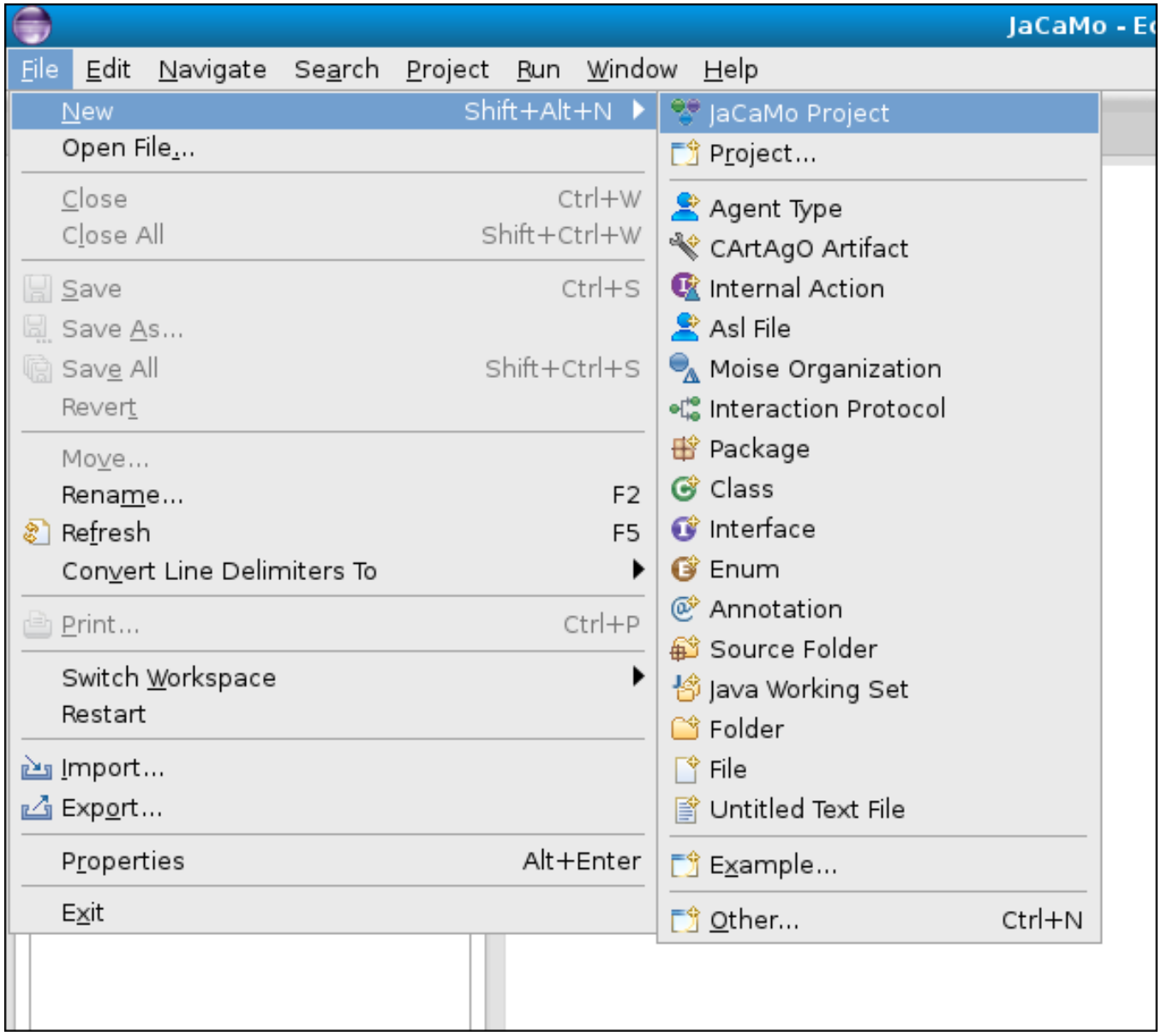
- Examples of codes, of demos
- Configuration of the platform (.jacamo file in the home directory):
  - to be done after each installation of the platform

JaCaMo multi-agent system development

- 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 projecName`: 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

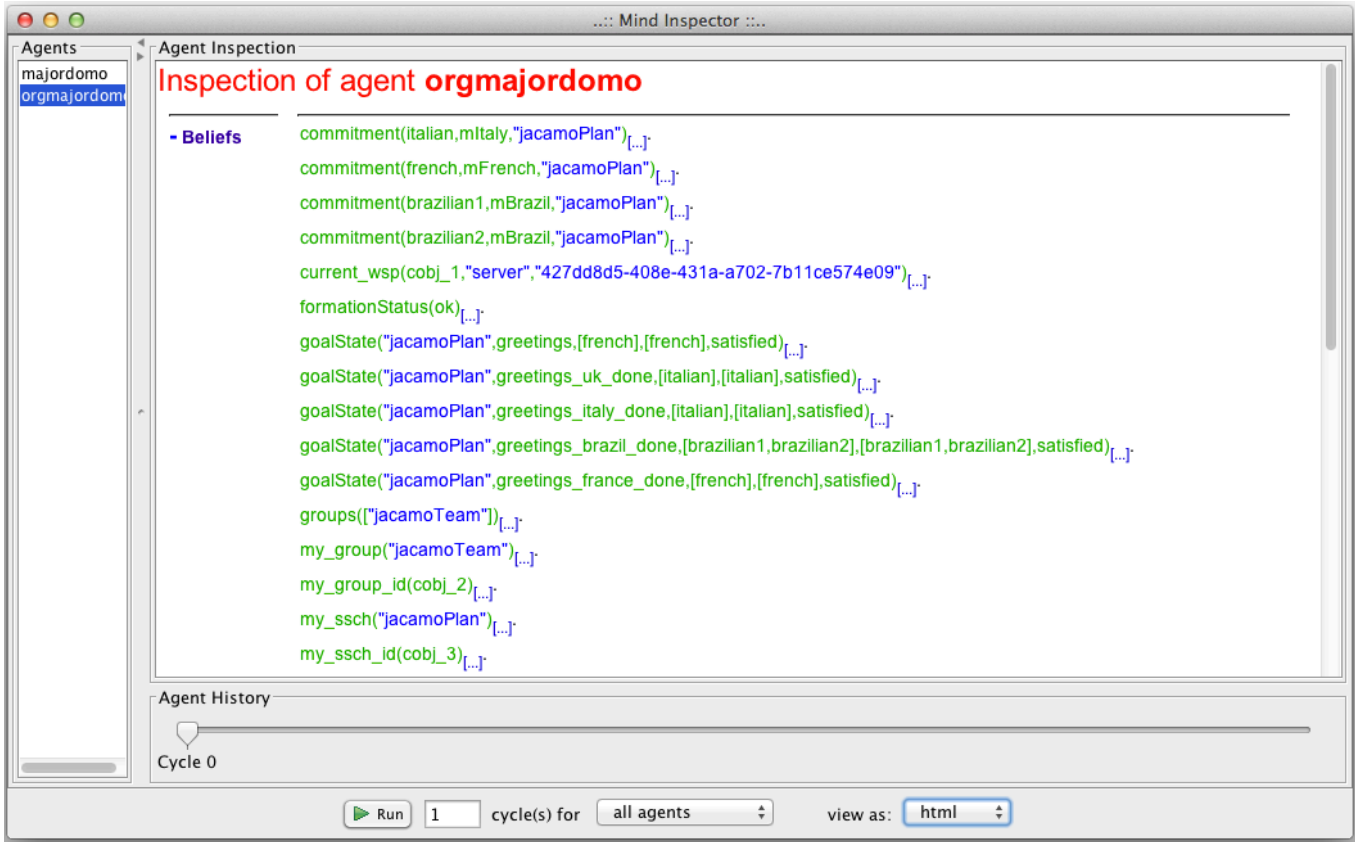
Eclipse JaCaMo plugin

JaCaMo multi-agent system development



Agent's inspector

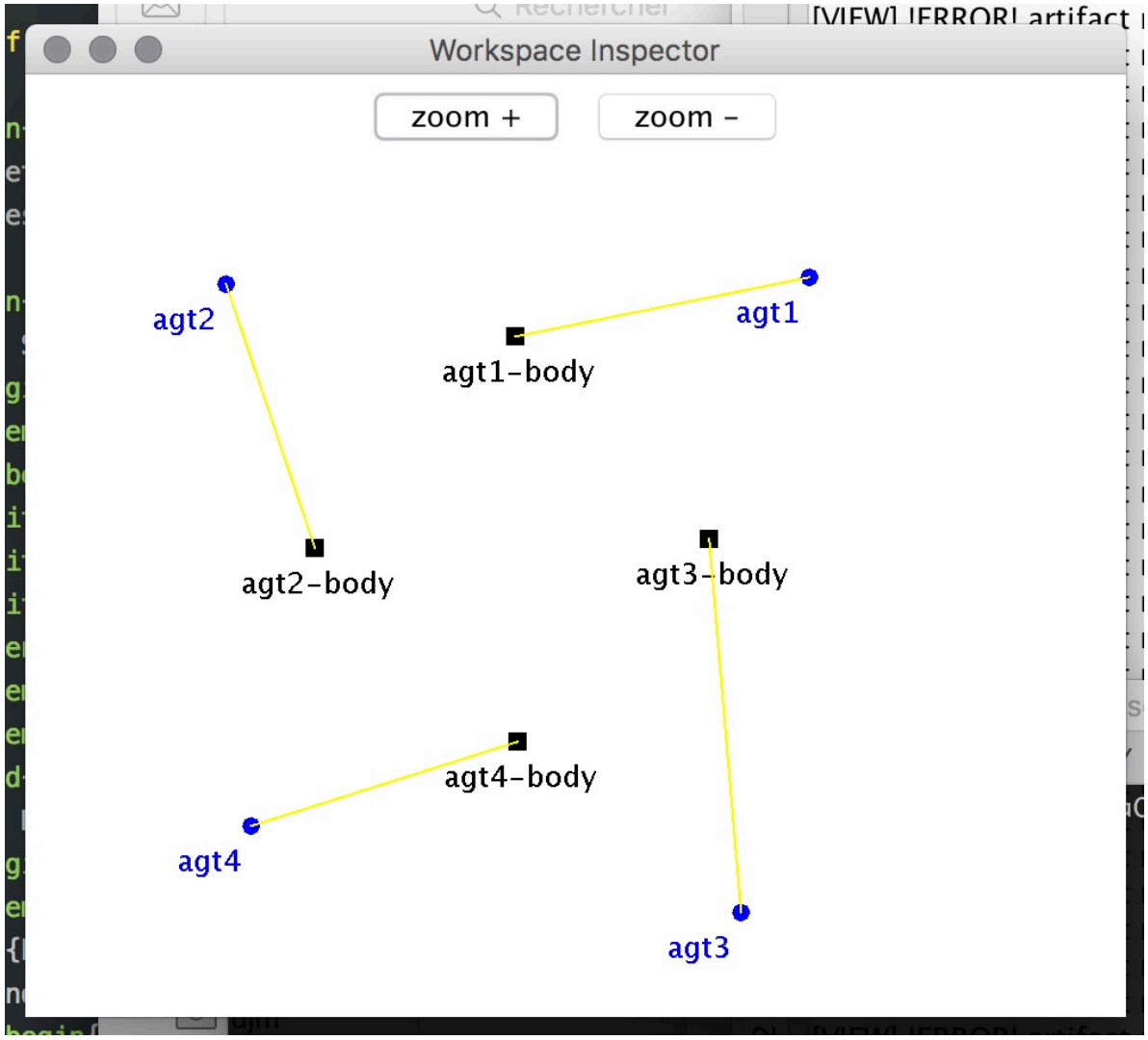
JaCaMo multi-agent system development



Runs also as an http server

Environment's inspector

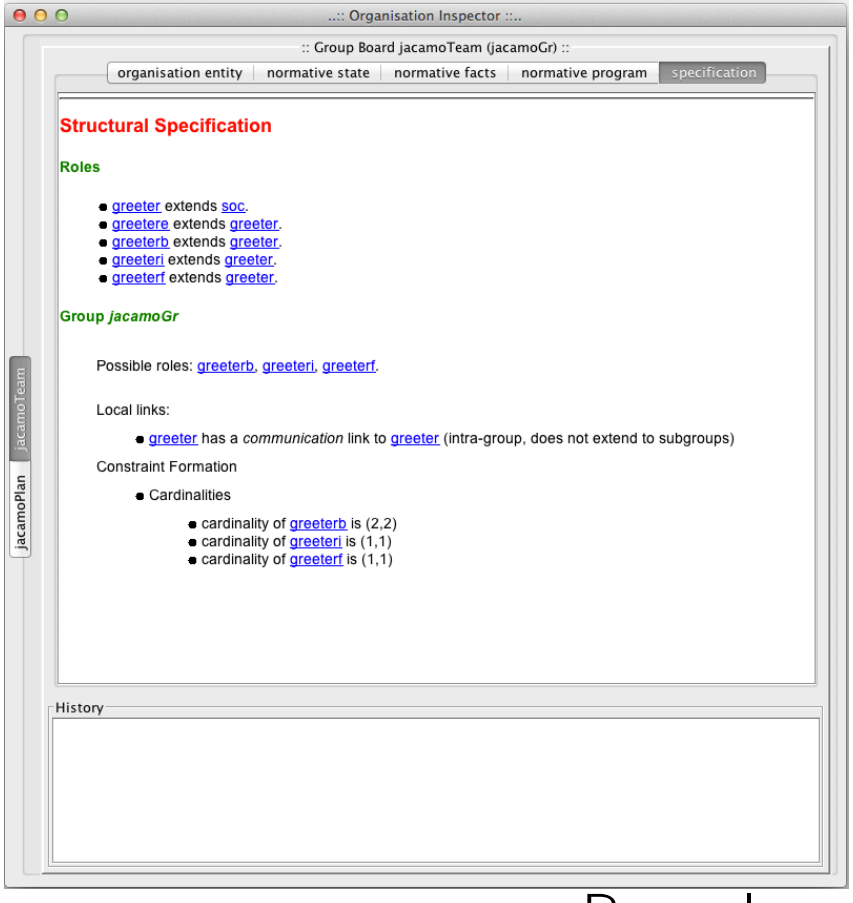
JaCaMo multi-agent system development

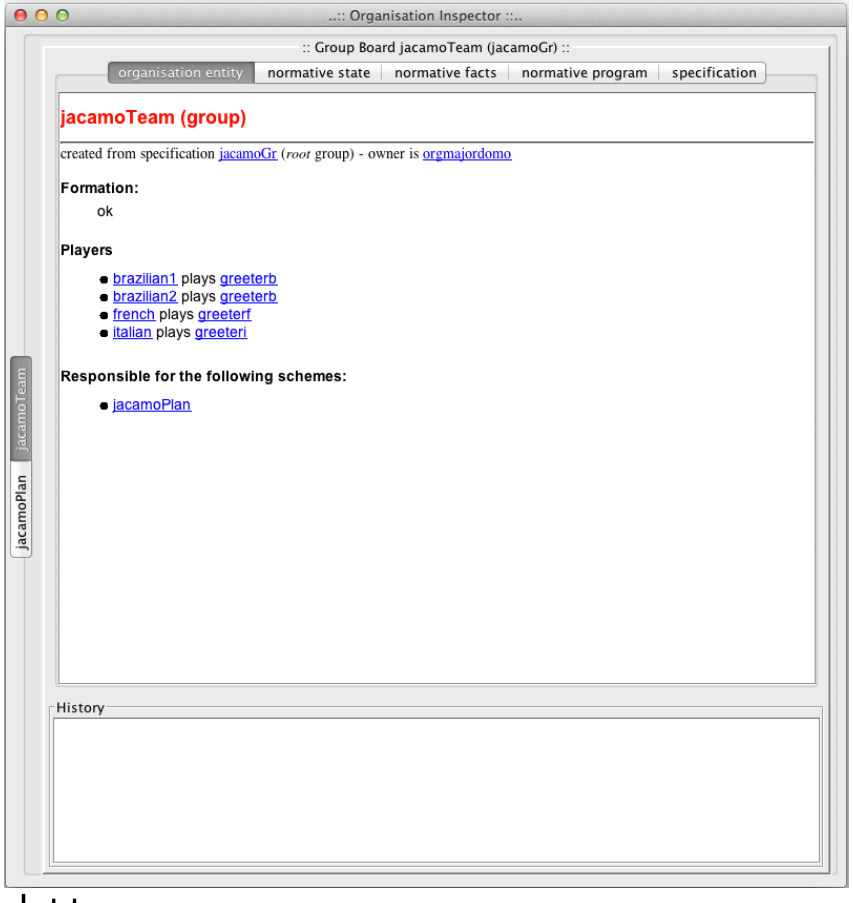


Runs also as an http server

Organization structure's inspector

JaCaMo multi-agent system development

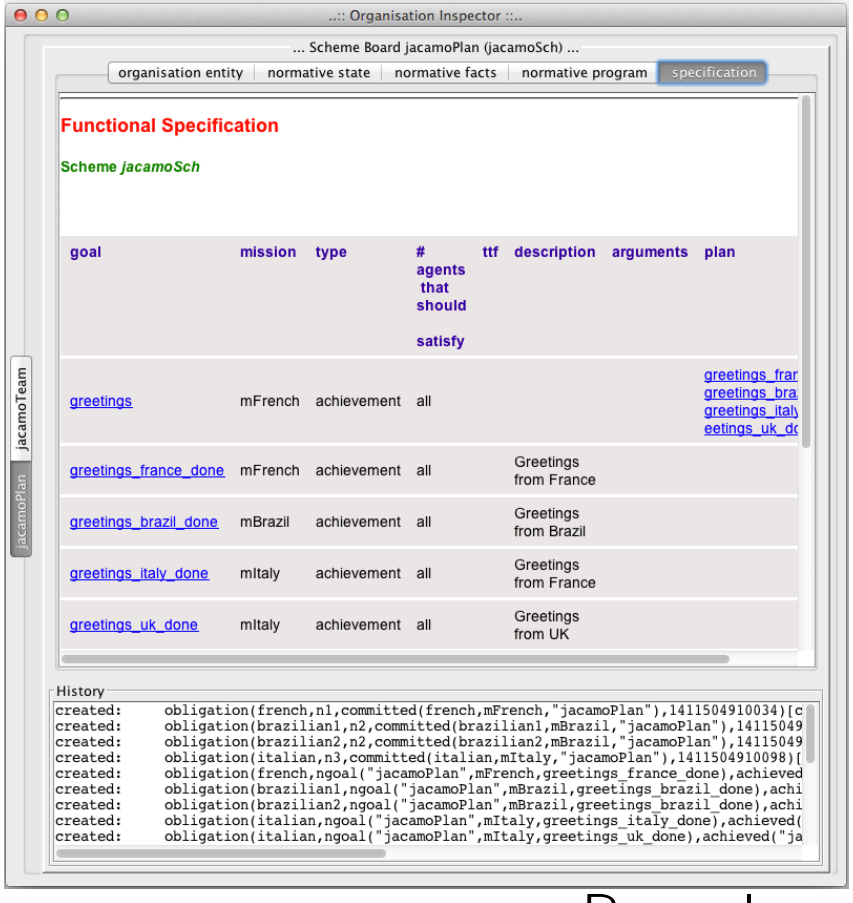


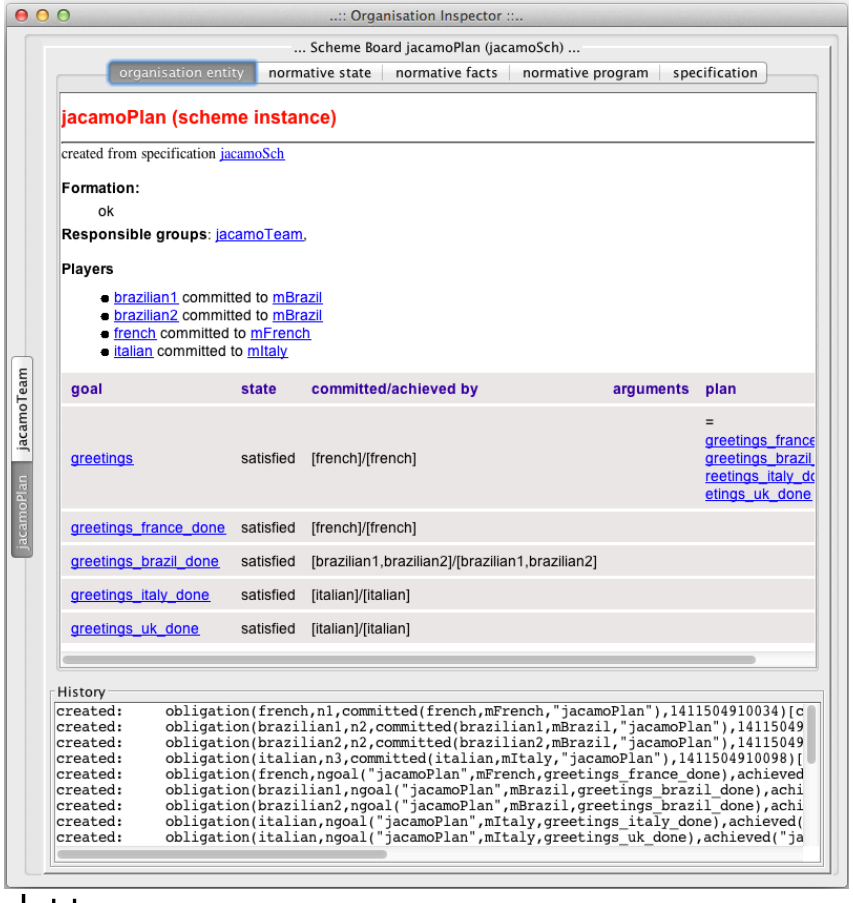


Runs also as an http server

Organization functioning's inspector

JaCaMo multi-agent system development





Runs also as an http server

Bibliography I

Aldewereld, H., Boissier, O., Dignum, V., Noriega, P., and Padget, J., editors (2016).

Social Coordination Frameworks for Social Technical Systems, volume 30 of *Law, Governance and Technology Series*.

Springer International Publishing.

Artikis, A., Sergot, M., Pitt, J., Busquets, D., and Riveret, R. (2016).

Specifying and Executing Open Multi-agent Systems, pages 197–212.

Springer International Publishing, Cham.

Bellifemine, F., Poggi, A., and Rimassa, G. (1999).

Jade–a fipa-compliant agent framework.

In *Proceedings of PAAM*, volume 99, page 33. London.

Bellifemine, F., Poggi, A., and Rimassa, G. (2000).

Developing multi-agent systems with jade.

In *International Workshop on Agent Theories, Architectures, and Languages*, pages 89–103. Springer.

Bibliography II

Boissier, O. (2003).

Contrôle et coordination orientés multi-agents.

Habilitation à diriger des recherches, ENS Mines Saint-Etienne et Université Jean Monnet.

Boissier, O., Bordini, R., Hübner, J. F., and Ricci, A. (2020).

Multi-Agent Oriented Programming: Programming Multi-Agent Systems Using JaCaMo.

The MIT Press.

Boissier, O., Bordini, R. H., Hübner, J. F., Ricci, A., and Santi, A. (2011).

Multi-agent oriented programming with jacamo.

Science of Computer Programming, pages –.

Bordini, R. H., Dastani, M., Dix, J., and Fallah-Seghrouchni, A. E., editors (2005).

Multi-Agent Programming: Languages, Platforms and Applications, volume 15 of *Multiagent Systems, Artificial Societies, and Simulated Organizations*.

Springer.

Bibliography III

Bordini, R. H., Hübner, J. F., and Wooldrige, M. (2007).

Programming Multi-Agent Systems in AgentSpeak using Jason.

Wiley Series in Agent Technology. John Wiley & Sons.

Campos, J., López-Sánchez, M., Rodriguez-Aguilar, J. A., and Esteva, M. (2009).

Formalising situatedness and adaptation in electronic institutions.

In *Coordination, Organizations, Institutions and Norms in Agent Systems IV*, volume 5428/2009 of *LNCS*. Springer Berlin / Heidelberg.

Cardoso, H. L., Urbano, J., Rocha, A. P., Castro, A. J., and Oliveira, E. (2016).

Ante: A framework integrating negotiation, norms and trust.

In *Social coordination frameworks for social technical systems*, pages 27–45. Springer.

Dastani, M., van Riemsdijk, M. B., Dignum, F., and Meyer, J.-J. C. (2003).

A programming language for cognitive agents goal directed 3apl.

In *International Workshop on Programming Multi-Agent Systems*, pages 111–130. Springer.

Bibliography IV

de Brito, M., Hübner, J. F., and Boissier, O. (2015).

Bringing constitutive dynamics to situated artificial institutions.

In *Proc. of 17th Portuguese Conference on Artificial Intelligence (EPIA 2015)*, volume 9273 of *LNCS*, pages 624–637. Springer.

Demazeau, Y. (1995).

From interactions to collective behaviour in agent-based systems.

In *Proc. of the 1st European Conf. on Cognitive Science. Saint-Malo*, pages 117–132.

García, E., Valero, S., and Giret, A. (2016).

ROMAS-Magentix2, pages 153–171.

Springer International Publishing, Cham.

Hindriks, K. V. (2009).

Programming rational agents in GOAL.

In Bordini, R. H., Dastani, M., Dix, J., and Fallah-Seghrouchni, A. E., editors, *Multi-Agent Programming*, pages 119–157. Springer.

Bibliography V

Howden, N., Rönnquist, R., Hodgson, A., and Lucas, A. (2001).

Jack intelligent agents-summary of an agent infrastructure.

In *5th International conference on autonomous agents*.

Hübner, J. F., Boissier, O., Kitio, R., and Ricci, A. (2009).

Instrumenting Multi-Agent Organisations with Organisational Artifacts and Agents.

Journal of Autonomous Agents and Multi-Agent Systems.

Huhns, M. N. (2001).

Interaction-oriented programming.

In *First international workshop, AOSE 2000 on Agent-oriented software engineering*, pages 29–44, Secaucus, NJ, USA. Springer-Verlag New York, Inc.

Lützenberger, M., Küster, T., Konnerth, T., Thiele, A., Masuch, N., Heßler, A., Keiser, J., Burkhardt, M., Kaiser, S., and Albayrak, S. (2013).

Jiac v: A mas framework for industrial applications.

In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pages 1189–1190. International Foundation for Autonomous Agents and Multiagent Systems.

Bibliography VI


Noriega, P. and de Jonge, D. (2016).
Electronic Institutions: The EI/EIDE Framework, pages 47–76.
Springer International Publishing, Cham.


Occello, M., Baeijs, C., Demazeau, Y., and Koning, J.-L. (2004).
MASK: An AEIO toolbox to design and build multi-agent systems.
In et al., C., editor, *Knowledge Engineering and Agent Technology*, IOS Series on Frontiers in AI and Applications. IOS press, Amsterdam.


Omicini, A., Ricci, A., and Viroli, M. (2008).
Artifacts in the A&A meta-model for multi-agent systems.
Autonomous Agents and Multi-Agent Systems, 17(3):432–456.


Padget, J., ElDeen Elakehal, E., Li, T., and De Vos, M. (2016).
InstAL: An Institutional Action Language, pages 101–124.
Springer International Publishing, Cham.


Piunti, M., Ricci, A., Boissier, O., and Hubner, J. (2009).
Embodying organisations in multi-agent work environments.
In *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT 2009)*, Milan, Italy.

Bibliography VII

Pokahr, A., Braubach, L., Haubeck, C., and Ladiges, J. (2014).

Programming bdi agents with pure java.

In *German Conference on Multiagent System Technologies*, pages 216–233. Springer.

Pokahr, A., Braubach, L., and Lamersdorf, W. (2005).

Jadex: A bdi reasoning engine.

In [Bordini et al., 2005], pages 149–174.

Pynadath, D. V., Tambe, M., Chauvat, N., and Cavedon, L. (1999).

Toward team-oriented programming.

In Jennings, N. R. and Lespérance, Y., editors, *ATAL*, volume 1757 of *LNCS*, pages 233–247. Springer.

Ricci, A., Piunti, M., and Viroli, M. (2010).

Environment programming in multi-agent systems – an artifact-based perspective.

Autonomous Agents and Multi-Agent Systems.

Published Online with ISSN 1573-7454 (will appear with ISSN 1387-2532).

Bibliography VIII

Ricci, A., Piunti, M., Viroli, M., and Omicini, A. (2009).

Environment programming in CArtAgO.

In *Multi-Agent Programming: Languages,Platforms and Applications,Vol.2*. Springer.

Ricordel, P. and Demazeau, Y. (2002).

VOLCANO: a vowels-oriented multi-agent platform.

In Dunin-Keplicz and Nawarecki, editors, *Proceedings of the International Conference of Central Eastern Europe on Multi-Agent Systems (CEEMAS'01)*, volume 2296 of *LNAI*, pages 252–262. Springer Verlag.

Rodriguez, S., Gaud, N., and Galland, S. (2014).

Sarl: a general-purpose agent-oriented programming language.

In *Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2014 IEEE/WIC/ACM International Joint Conferences on*, volume 3, pages 103–110. IEEE.

Shoham, Y. (1993).

Agent-oriented programming.

Artif. Intell., 60(1):51–92.

Bibliography IX

Sierhuis, M., Bradshaw, J. M., Acquisti, A., Van Hoof, R., Jeffers, R., and Uszok, A. (2003).

Human-agent teamwork and adjustable autonomy in practice.

In *Proceedings of the seventh international symposium on artificial intelligence, robotics and automation in space (I-SAIRAS)*.

Stratulat, T., Ferber, J., and Tranier, J. (2009).

MASQ: towards an integral approach to interaction.

In *AAMAS (2)*, pages 813–820.

Tampitsikas, C., Bromuri, S., and Schumacher, M. I. (2011).

MANET: A model for first-class electronic institutions.

In Cranefield, S., van Riemsdijk, M. B., Vázquez-Salceda, J., and Noriega, P., editors, *Coordination, Organizations, Institutions, and Norms in Agent System VII, COIN 2011 International Workshops, COIN@AAMAS 2011, Taipei, Taiwan, May 3, 2011, COIN@WI-IAT 2011, Lyon, France, August 22, 2011, Revised Selected Papers*, Lecture Notes in Computer Science, pages 75–92. Springer.

Winikoff, M. (2005).

Jack intelligent agents: An industrial strength platform.

In [Bordini et al., 2005], pages 175–193.