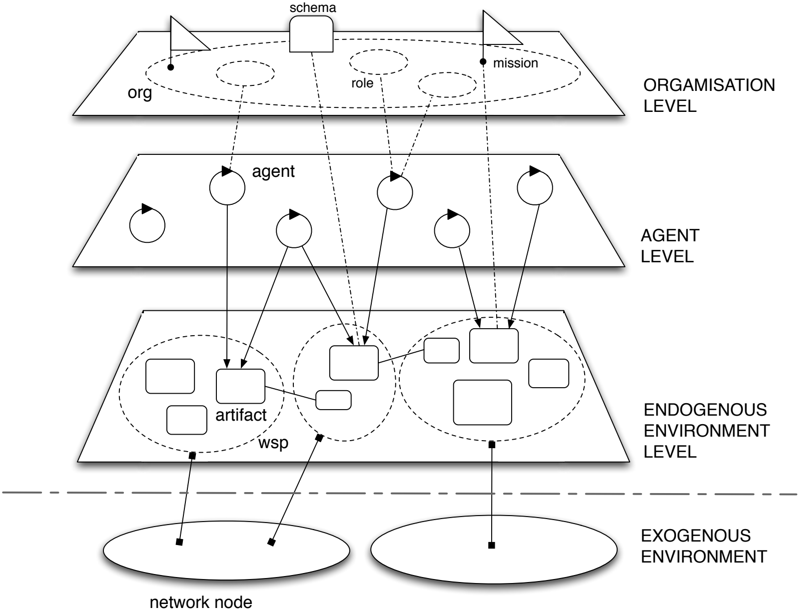
Organisation Oriented Programming with $\mathcal{M}$oise

Jomi F. Hübner

Federal University of Santa Catarina, Brazil

PPGEAS 2017 — UFSC

Multi-Agent System (our perspective)



Outline

- Fundamentals
- Approches
- $\mathcal{M}$oise (language)
- ORA4MAS (platform)
- Conclusions

(slides written together with O. Boissier, A. Ricci, and R. Bordini)

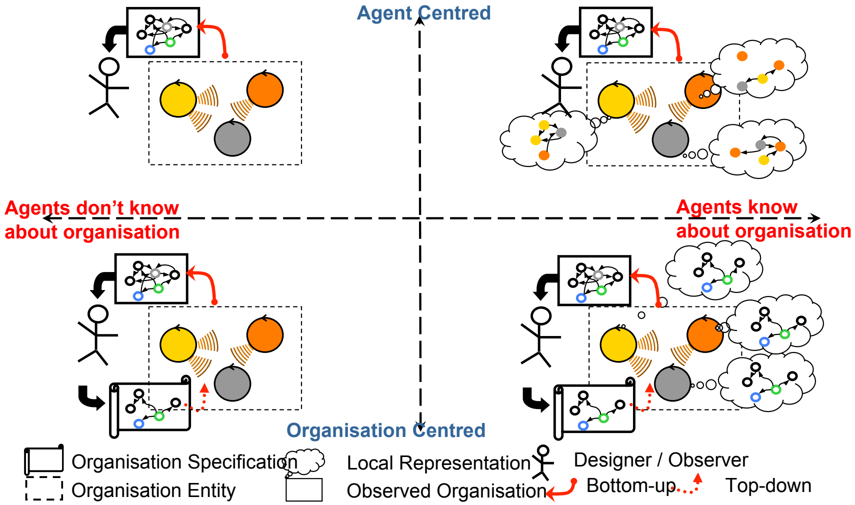
Introduction: Some definitions

- Organisations are structured, patterned systems of activity, knowledge, culture, memory, history, and capabilities that are distinct from any single agent [Gasser, 2001]
  ⇝ organisations are supra-individual phenomena
- A decision and communication schema which is applied to a set of actors that together fulfill a set of tasks in order to satisfy goals while guarantying a global coherent state [Malone, 1999]
  ⇝ definition by the designer, or by actors, to achieve a purpose
- An organisation is characterised by: a division of tasks, a distribution of roles, authority systems, communication systems, contribution-retribution systems [Bernoux, 1985]
  ⇝ pattern of predefined cooperation
- An arrangement of relationships between components, which results into an entity, a system, that has unknown skills at the level of the individuals [Morin, 1977]
  ⇝ pattern of emergent cooperation

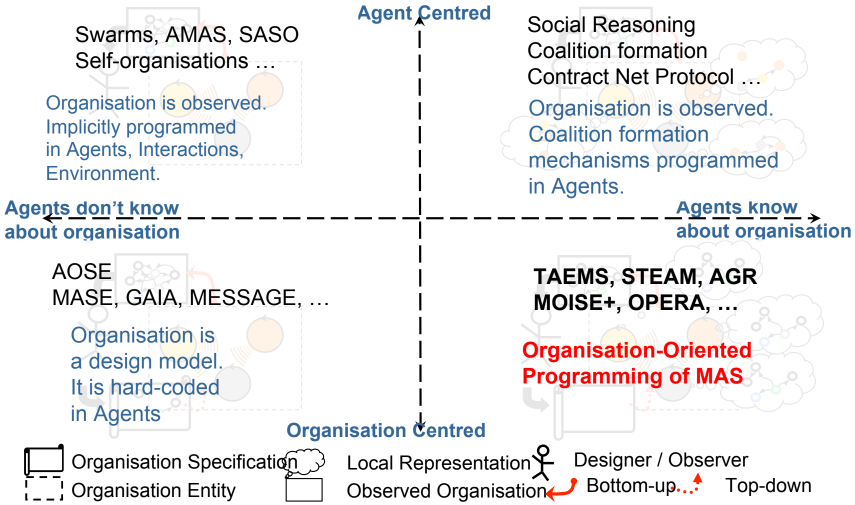
Organisation in MAS – a definition

- Pattern of agent cooperation
  - with a purpose
  - supra-agent
  - emergent or
  - predefined (by designer or agents)

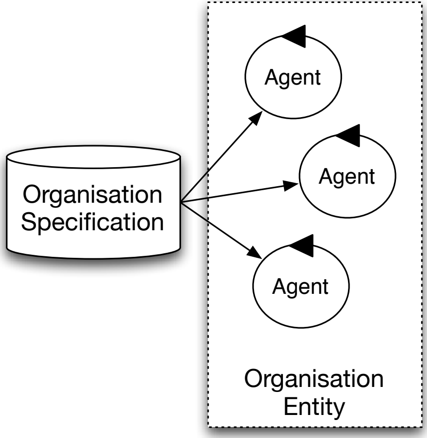
Perspective on organisations from EASSS'05 Tutorial (Sichman, Boissier)



Perspective on organisations from EASSS'05 Tutorial (Sichman, Boissier)



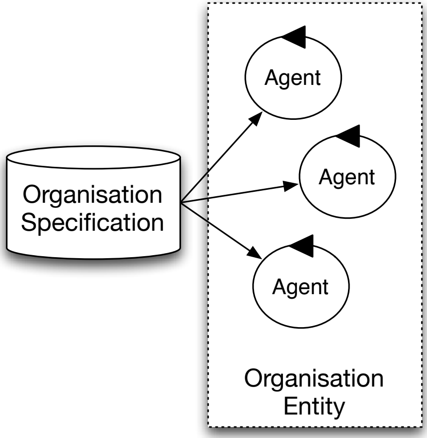
Organisation Oriented Programming (OOP)



- Programming outside the agents
- Using organisational concepts
- To define a cooperative pattern

- Program = Specification
- By changing the specification, we can change the MAS overall behaviour

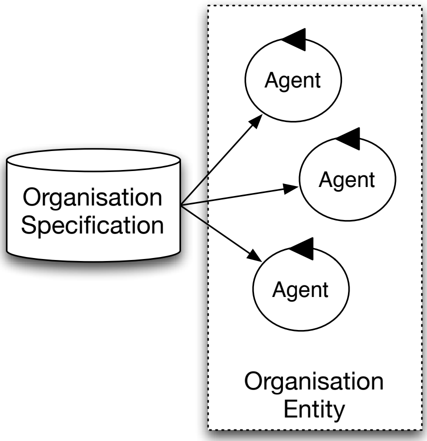
Organisation Oriented Programming (OOP)



First approach

- Agents read the program and follow it

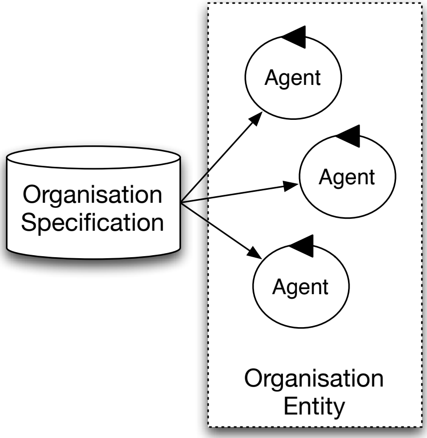
Organisation Oriented Programming (OOP)



Second approach

- Agents are forced to follow the program
- Agents are rewarded if they follow the program
- ...

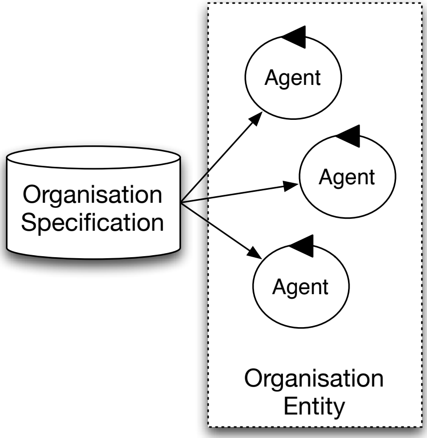
Organisation Oriented Programming (OOP)



Second approach

- Agents are forced to follow the program
- Agents are rewarded if they follow the program
- ...

Organisation Oriented Programming (OOP)



Components

- Programming language (OML)
- Platform (OMI)
- Integration to agent architectures and to environment

Components of OOP: Organisation Modelling Language (OML)

- Declarative specification of the organisation(s)
- Specific constraints, norms and cooperation patterns imposed on the agents
- Based on an organisational model

  e.g. AGR [Ferber and Gutknecht, 1998],
  TeamCore [Tambe, 1997],
  Islander [Esteva et al., 2001],
  $\mathcal{M}$oise^{+} [Hübner et al., 2002],
  Opera [Dignum and Aldewereld, 2010],
  2OPL [Dastani et al., 2009],
  THOMAS [Criado et al., 2011],
  ...

Components of OOP: Organisation Management Infrastructure (OMI)

- Coordination mechanisms, i.e. support infrastructure
  - e.g. MadKit [Gutknecht and Ferber, 2000],
    karma [Pynadath and Tambe, 2003],
    ...
- Regulation mechanisms, i.e. governance infrastructure
  - e.g. Ameli [Esteva et al., 2004],
    $\mathcal{S}$-$\mathcal{M}$oise^{+} [Hübner et al., 2006],
    ORA4MAS [Hübner et al., 2009],
    ...
- Adaptation mechanisms, i.e. reorganisation infrastructure

Components of OOP: Integration mechanisms

- Agent integration mechanisms
  allow agents to be aware of and to deliberate on:
  - entering/exiting the organisation
  - modification of the organisation
  - obedience/violation of norms
  - sanctioning/rewarding other agents

  e.g. $\mathcal{J}$-$\mathcal{M}$oise$^+$ [Hübner et al., 2007], Autonomy based reasoning [Carabelea, 2007], *ProsA*$_2$ Agent-based reasoning on norms [Ossowski, 1999], ...
- Environment integration mechanisms
  transform organisation into embodied organisation so that:
  - organisation may act on the environment (e.g. enact rules, regimentation)
  - environment may act on the organisation (e.g. count-as rules)

  e.g [Piunti et al., 2009b, Okuyama et al., 2008, de Brito et al., 2014]

Motivations for OOP: **Applications** point of view

- Current applications show an increase in
  - Number of agents
  - Duration and repetitiveness of agent activities
  - Heterogeneity of the agents
  - Number of designers of agents
  - Agent ability to act and decide
  - Openness, scalability, dynamism
- More and more applications require the integration of human communities and technological communities (ubiquitous and pervasive computing), building connected communities (ICities) in which agents act on behalf of users
  - Trust, security, ..., flexibility, adaptation

Motivations for OOP: **Constitutive** point of view

- Organisation helps the agents to cooperate with other agents by defining common cooperation schemes
  - global tasks
  - protocols
  - groups, responsibilities

e.g. 'to bid' for a product on eBay is an institutional action only possible because eBay defines rules for that very action
  - the bid protocol is a constraint but it also creates the action

e.g. when a soccer team wants to play match, the organisation helps the members of the team to synchronise actions, to share information, etc

Motivations for OOP: **Normative** point of view

- MAS have two properties which seem contradictory:
  - a global purpose
  - autonomous agents
  - ⇝ While the autonomy of the agents is essential, it may cause loss in the global coherence of the system and achievement of the global purpose
- Embedding norms within the organisation of an MAS is a way to constrain the agents' behaviour towards the global purposes of the organisation, while explicitly addressing the autonomy of the agents within the organisation
  - ⇝ Normative organisation
  - e.g. when an agent adopts a role, it adopts a set of behavioural constraints that support the global purpose of the organisation.
    It may decide to obey or disobey these constraints

Motivations for OOP: **Agents** point of view

An organisational specification is required to enable agents to "reason" about the organisation:

- to decide to enter into/leave from the organisation during execution
  - ⇝ Organisation is no more closed
- to change/adapt the current organisation
  - ⇝ Organisation is no more static
- to obey/disobey the organisation
  - ⇝ Organisation is no more a regimentation

Motivations for OOP: **Organisation** point of view

An organisational specification is required to enable the organisation to "reason" about itself and about the agents in order to ensure the achievement of its global purpose:

- to decide to let agents enter into/leave from the organisation during execution
  - ⇝ Organisation is no more closed
- to decide to let agents change/adapt the current organisation
  - ⇝ Organisation is no more static and blind
- to govern agents behaviour in the organisation (i.e. monitor, enforce, regiment)
  - ⇝ Organisation is no more a regimentation

Some OOP approaches

- AGR/Madkit [Ferber and Gutknecht, 1998]
- STEAM/Teamcore [Tambe, 1997]
- ISLANDER/AMELI [Esteva et al., 2004]
- Opera/Operetta [Dignum and Aldewereld, 2010]
- PopOrg [Rocha Costa and Dimuro, 2009]
- 2OPL [Dastani et al., 2009]
- THOMAS [Criado et al., 2011],
- ...

Fundamentals

Some OOP approaches

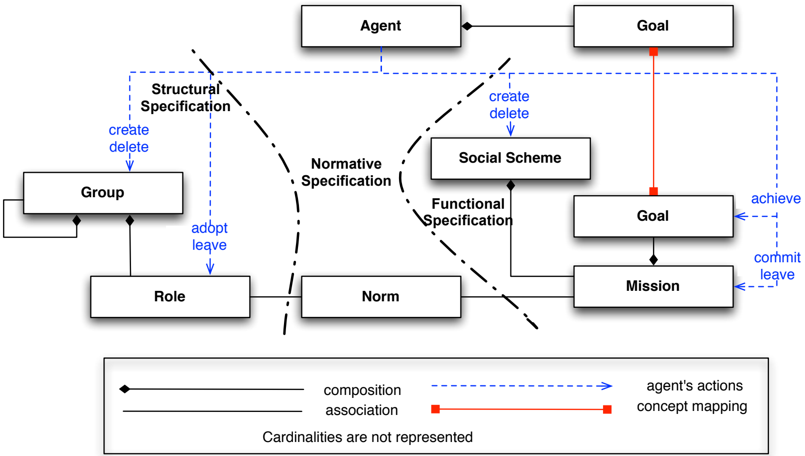
The $\mathcal{M}$oise framework

- $\mathcal{M}$oise Organisation Modelling Language (OML)
- ORA4MAS Organisation Management Infrastructure (OMI)
- Jason and ORA4MAS integration

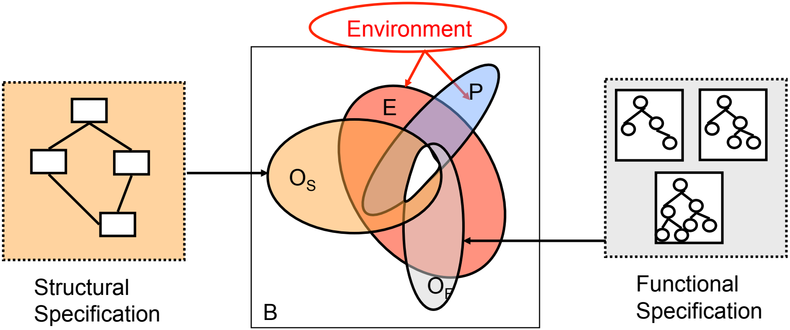
$\mathcal{M}$oise Framework

- OML (language)
  - Tag-based language
    (issued from $\mathcal{M}$oise [Hannoun et al., 2000],
    $\mathcal{M}$oise$^+$ [Hübner et al., 2002],
    $\mathcal{M}$oiseInst [Gâteau et al., 2005])
- OMI (infrastructure)
  - developed as an artifact-based working environment
    (ORA4MAS [Hübner et al., 2009] based on CArtAgO nodes,
    refactoring of $\mathcal{S}$-$\mathcal{M}$oise$^+$ [Hübner et al., 2006] and
    $\mathcal{S}$ynai [Gâteau et al., 2005])
- Integrations
  - Agents and Environment (c4Jason, c4Jadex
    [Ricci et al., 2009])
  - Environment and Organisation ([Piunti et al., 2009a])
  - Agents and Organisation ($\mathcal{J}$-$\mathcal{M}$oise$^+$ [Hübner et al., 2007])

$\mathcal{M}$oise OML meta-model (partial view)



$\mathcal{M}$oise Modelling Dimensions



Structural Specification

Functional Specification

Groups, links, roles
Compatibilities, multiplicities
inheritance

Normative Specification
Permissions, Obligations
Allows agents autonomy!

Global goals, plans,
Missions, schemas,
preferences

$\mathcal{M}$oise OML

- OML for defining organisation specification and organisation entity
- Three independent dimensions [Hübner et al., 2007] ($\rightsquigarrow$ well adapted for the reorganisation concerns):
  - Structural: Roles, Groups
  - Functional: Goals, Missions, Schemes
  - Normative: Norms (obligations, permissions, interdictions)
- Abstract description of the organisation for
  - the designers
  - the agents

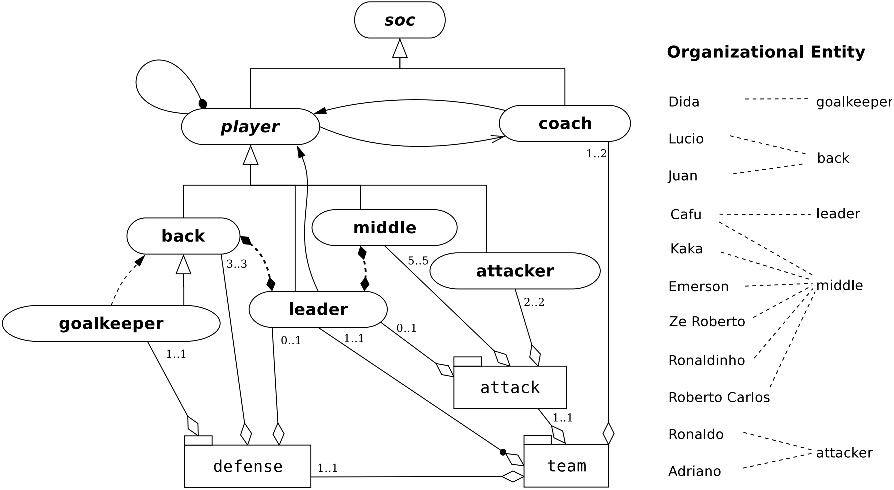
    $\rightsquigarrow$ $\mathcal{J}$-$\mathcal{M}$oise^{+} [Hübner et al., 2007]
  - the Organisation Management Infrastructure

    $\rightsquigarrow$ ORA4MAS [Hübner et al., 2009]

Structural Specification

- Specifies the structure of an MAS along three levels:
  - Individual with Role
  - Social with Link
  - Collective with Group
- Components:
  - Role: label used to assign rights and constraints on the behavior of agents playing it
  - Link: relation between roles that directly constrains the agents in their interaction with the other agents playing the corresponding roles
  - Group: set of links, roles, compatibility relations used to define a shared context for agents playing roles in it

Structural Specification Example

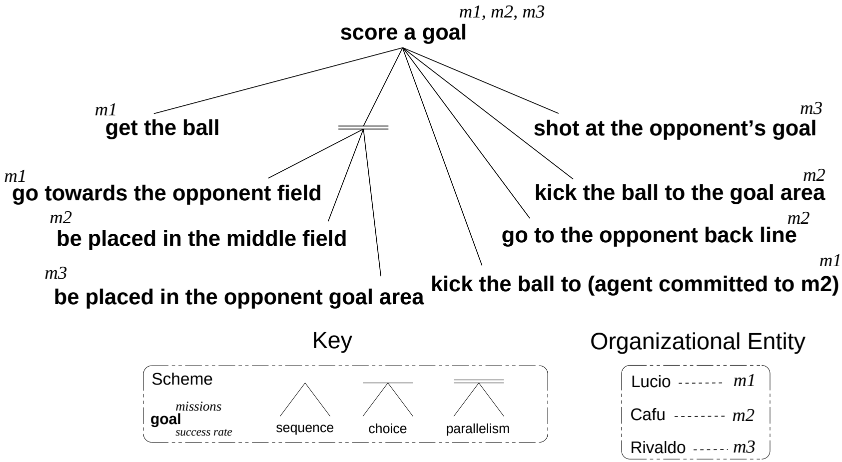


Graphical representation of structural specification of 3-5-2 Joj Team

Functional Specification

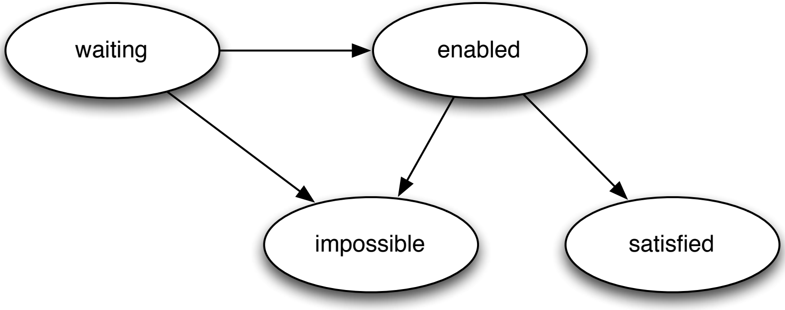
- Specifies the expected behaviour of an MAS in terms of goals along two levels:
  - Collective with Scheme
  - Individual with Mission
- Components:
  - Goals:
    - Achievement goal (default type). Goals of this type should be declared as satisfied by the agents committed to them, when achieved
    - Maintenance goal. Goals of this type are not satisfied at a precise moment but are pursued while the scheme is running. The agents committed to them do not need to declare that they are satisfied
  - Scheme: global goal decomposition tree assigned to a group
    - Any scheme has a root goal that is decomposed into subgoals
  - Missions: set of coherent goals assigned to roles within norms

Functional Specification Example



Graphical representation of social scheme "side_attack" for joj team

Goal States



waiting	initial state
enabled	goal pre-conditions are satisfied & scheme is well-formed
satisfied	agents committed to the goal have achieved it
impossible	the goal is impossible to be satisfied

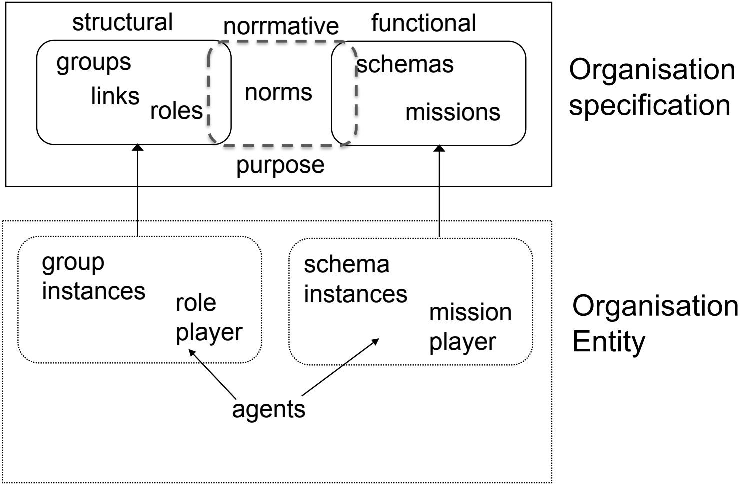
Normative Specification

- Explicit relation between the functional and structural specifications
- Permissions and obligations to commit to missions in the context of a role
- Makes explicit the normative dimension of a role

Norm Specification – example

role	**deontic**	**mission**		TTF
back	*obliged*	$m1$	get the ball, go ...	1 minute
left	*obliged*	$m2$	be placed at ..., kick ...	3 minute
right	*obliged*	$m2$		1 day
attacker	*obliged*	$m3$	kick to the goal, ...	30 seconds

Organisational Entity



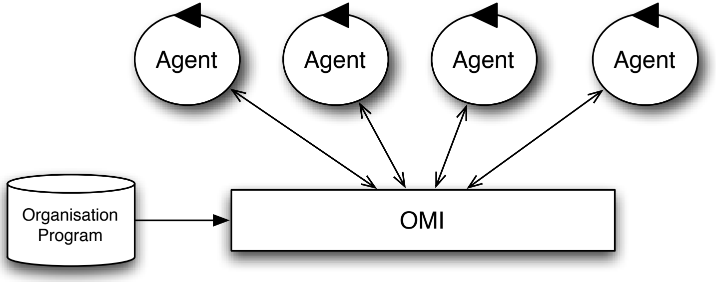
Organisation Entity Dynamics

1. Organisation is created (by the agents)
   - instances of groups
   - instances of schemes
2. Agents enter into groups adopting roles
3. Groups become responsible for schemes
   - Agents from the group are then obliged to commit to missions in the scheme
4. Agents commit to missions
5. Agents fulfil mission's goals
6. Agents leave schemes and groups
7. Schemes and groups instances are destroyed

Organisation management infrastructure (OMI)

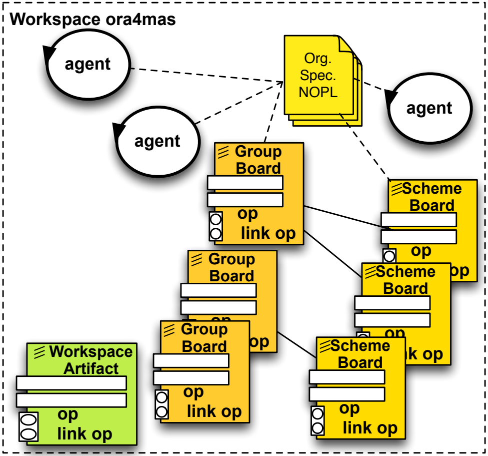
Responsibility

- Managing – coordination, regulation – the agents' execution within organisation defined in an organisational specification



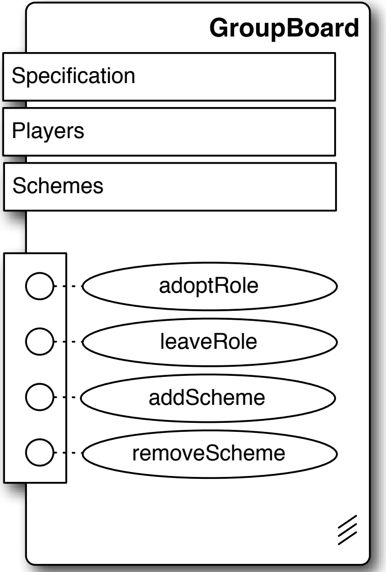
(e.g. MadKit, AMELI, $\mathcal{S}$-$\mathcal{M}$oise^{+}, THOMAS, ...)

Organisational artifacts in ORA4MAS



- based on A&A and $\mathcal{M}$oise
- agents create and handle organisational artifacts
- artifacts in charge of regimentations, detection and evaluation of norms compliance
- agents are in charge of decisions about sanctions
- distributed solution

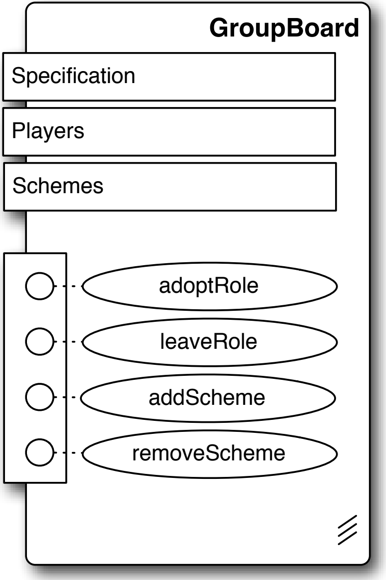
ORA4MAS – GroupBoard artifact



Observable Properties:

- specification: the specification of the group in the OS (an object of class moise.os.ss.Group)
- players: a list of agents playing roles in the group. Each element of the list is a pair (agent x role)
- schemes: a list of scheme identifiers that the group is responsible for

ORA4MAS – GroupBoard artifact



Operations:

- adoptRole(role): the agent executing this operation tries to adopt a role in the group
- leaveRole(role)
- addScheme(schid): the group starts to be responsible for the scheme managed by the SchemeBoard schId
- removeScheme(schid)

ORA4MAS – SchemeBoard artifact

Observable Properties:

- specification: the specification of the scheme in the OS
- groups: a list of groups responsible for the scheme
- players: a list of agents committed to the scheme. Each element of the list is a pair (agent, mission)
- goals: a list with the current state of the goals
- obligations: list of obligations currently active in the scheme

ORA4MAS – SchemeBoard artifact

Operations:

- commitMission(mission) and leaveMission: operations to "enter" and "leave" the scheme
- goalAchieved(goal): defines that some goal is achieved by the agent performing the operation
- setGoalArgument(goal, argument, value): defines the value of some goal's argument

Environment integration: constitutive rules

Count-As rule

An event occurring on an artifact, in a particular context, may count-as an institutional event

$\rightsquigarrow$ indirect automatic updating of the organisation

Agent integration

- Agents can interact with organisational artifacts as with ordinary artifacts by perception and action

$\leadsto$ Any Agent Programming Language integrated with CArtAgO can use organisational artifacts

Agent integration provides some "internal" tools for the agents to simplify their interaction with the organisation:

- maintenance of a local copy of the organisational state
- production of organisational events
- provision of organisational actions

Organisational **actions** in *Jason* I

Example (GroupBoard)

```
...
joinWorkspace("ora4mas",O4MWsp);
makeArtifact(
    "auction",
    "ora4mas.nopl.GroupBoard",
    ["auction-os.xml", auctionGroup],
    GrArtId);
adoptRole(auctioneer);
focus(GrArtId);
...
```

Organisational **actions** in *Jason* II

Example (SchemeBoard)

```
...
makeArtifact(
   "sch1",
   "ora4mas.nopl.SchemeBoard",
   ["auction-os.xml", doAuction],
   SchArtId);
focus(SchArtId);
addScheme(Sch);
commitMission(mAuctioneer)[artifact_id(SchArtId)];
...
```

Organisational perception

When an agent focus on an Organisational Artifact, the observable properties (Java objects) are translated to beliefs with the following predicates:

- specification
- play(agent, role, group)
- commitment(agent, mission, scheme)
- goalState(scheme, goal, list of committed agents, list of agent that achieved the goal, state of the goal)
- obligation(agent,norm,goal,dead line)
- ....

Organisational perception – example

Inspection of agent **bob** (cycle #0)

- Beliefs

commitment(bob,mManager,"sch2")[artifact_id(cobj_4),c
cept),artifact_name(cobj_4,"sch2"),artifact_type(cobj_4,"ora4m
commitment(bob,mManager,"sch1")[artifact_id(cobj_3),c
cept),artifact_name(cobj_3,"sch1"),artifact_type(cobj_3,"ora4m
current_wsp(cobj_1,"ora4mas","308b05b0-2994-4fe8
formationStatus(ok)[artifact_id(cobj_2),obs_prop_id("obs_i
obj_2,"mypaper"),artifact_type(cobj_2,"ora4mas.nopl.GroupBo
goalState("sch2",wp,[bob],[bob],satisfied)[artifact_id(cob

Handling organisational **events** in *Jason*

Whenever something changes in the organisation, the agent architecture updates the agent belief base accordingly producing events (belief update from perception)

Example (new agent entered the group)

```
+play(Ag,boss,GId) <- .send(Ag,tell,hello).
```

Example (change in goal state and norm violation)

```
+goalState(Scheme,wsecs,_,_,satisfied)
    : .my_name(Me) & commitment(Me,mCol,Scheme)
   <- leaveMission(mColaborator,Scheme).

+normFailure(N) <- .print("norm failure event: ", N).
```

Typical plans for obligations

```
+obligation(Ag,Norm,committed(Ag,Mission,Scheme),DeadLine)
    : .my_name(Ag)
   <- .print("I am obliged to commit to ",Mission);
      commitMission(Mission,Scheme).

+obligation(Ag,Norm,achieved(Sch,Goal,Ag),DeadLine)
    : .my_name(Ag)
   <- .print("I am obliged to achieve goal ",Goal);
      !Goal[scheme(Sch)];
      goalAchieved(Goal,Sch).

+obligation(Ag,Norm,What,DeadLine)
   : .my_name(Ag)
   <- .print("I am obliged to ",What,
             ", but I don't know what to do!").
```

Summary – OOP

- Organisation is a complex and rich dimension in MAS:
  - represented in different "eyes":
    designer – observer – agents
  - expressed with two points of view:
    agent-centred vs. organisation-centred
  - using multiple models: e.g. Joint intentions, shared plans, dependence theory, ...
- Organisation is built to fulfill different aims
  - to help the cooperation between the agents,
  - to control the cooperation between the agents
    (forgetting or not the autonomy of the agents)

Summary – *M*oise

- Ensures that the agents follow some of the constraints specified for the organisation
- Helps the agents to work together
- The organisation is interpreted at runtime, it is not hardwired in the agents code
- The agents 'handle' the organisation (i.e. their artifacts)
- It is suitable for open systems as no specific agent architecture is required

- All available as open source at

`http://moise.souceforge.net`

TOC

Bibliography I

Bernoux, P. (1985).
La sociologie des organisations.
Seuil, 3ème edition.

Carabelea, C. (2007).
Reasoning about autonomy in open multi-agent systems - an approach based on the social power theory.
in french, ENS Mines Saint-Etienne.

Criado, N., Argente, E., and Botti, V. (2011).
THOMAS: An agent platform for supporting normative multi-agent systems.
Journal of Logic and Computation, 23(2):309–333.

Dastani, M., Grossi, D., Meyer, J.-J., and Tinnemeier, N. (2009).
Normative multi-agent programs and their logics.
In Meyer, J.-J. and Broersen, J., editors, *Knowledge Representation for Agents and Multi-Agent Systems*, volume 5605 of *Lecture Notes in Computer Science*, pages 16–31. Springer Berlin / Heidelberg.

Bibliography II

de Brito, M., Hübner, J. F., and Boissier, O. (2014).
A conceptual model for situated artificial institutions.
In Bulling, N., van der Torre, L. W. N., Villata, S., Jamroga, W., and Vasconcelos, W., editors, *Computational Logic in Multi-Agent Systems - 15th International Workshop, CLIMA XV, Prague, Czech Republic, August 18-19, 2014. Proceedings*, volume 8624 of *Lecture Notes in Computer Science*, pages 35–51. Springer.

Dignum, V. and Aldewereld, H. (2010).
Operetta: Organization-oriented development environment.
In *Proceedings of LADS @ MALLOW 2010*, pages 14–20.

Esteva, M., Rodriguez-Aguiar, J. A., Sierra, C., Garcia, P., and Arcos, J. L. (2001).
On the formal specification of electronic institutions.
In Dignum, F. and Sierra, C., editors, *Proceedings of the Agent-mediated Electronic Commerce*, LNAI 1191, pages 126–147, Berlin. Springer.

Esteva, M., Rodríguez-Aguilar, J. A., Rosell, B., and L., J. (2004).
AMELI: An agent-based middleware for electronic institutions.
In Jennings, N. R., Sierra, C., Sonenberg, L., and Tambe, M., editors, *Proc. of the 3rd Int. Joint Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS'04)*, pages 236–243, New York, USA. ACM.

Bibliography III

Ferber, J. and Gutknecht, O. (1998).
A meta-model for the analysis and design of organizations in multi-agents systems.
In Demazeau, Y., editor, *Proceedings of the 3rd International Conference on Multi-Agent Systems (ICMAS'98)*, pages 128–135. IEEE Press.

Gasser, L. (2001).
Organizations in multi-agent systems.
In *Pre-Proceeding of the 10th European Worshop on Modeling Autonomous Agents in a Multi-Agent World (MAAMAW'2001)*, Annecy.

Gâteau, B., Boissier, O., Khadraoui, D., and Dubois, E. (2005).
Moiseinst: An organizational model for specifying rights and duties of autonomous agents.
In *Third European Workshop on Multi-Agent Systems (EUMAS 2005)*, pages 484–485, Brussels Belgium.

Gutknecht, O. and Ferber, J. (2000).
The MadKit agent platform architecture.
In *Agents Workshop on Infrastructure for Multi-Agent Systems*, pages 48–55.

Bibliography IV

Hannoun, M., Boissier, O., Sichman, J. S., and Sayettat, C. (2000).
Moise: An organizational model for multi-agent systems.
In Monard, M. C. and Sichman, J. S., editors, *Proceedings of the International Joint Conference, 7th Ibero-American Conference on AI, 15th Brazilian Symposium on AI (IBERAMIA/SBIA'2000), Atibaia, SP, Brazil, November 2000*, LNAI 1952, pages 152–161, Berlin. Springer.

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.

Hübner, J. F., Sichman, J. S., and Boissier, O. (2002).
A model for the structural, functional, and deontic specification of organizations in multiagent systems.
In Bittencourt, G. and Ramalho, G. L., editors, *Proceedings of the 16th Brazilian Symposium on Artificial Intelligence (SBIA'02)*, volume 2507 of *LNAI*, pages 118–128, Berlin. Springer.

Bibliography V

Hübner, J. F., Sichman, J. S., and Boissier, O. (2006).
S-MOISE+: A middleware for developing organised multi-agent systems.
In Boissier, O., Dignum, V., Matson, E., and Sichman, J. S., editors, *Coordination, Organizations, Institutions, and Norms in Multi-Agent Systems*, volume 3913 of *LNCS*, pages 64–78. Springer.

Hübner, J. F., Sichman, J. S., and Boissier, O. (2007).
Developing Organised Multi-Agent Systems Using the MOISE+ Model: Programming Issues at the System and Agent Levels.
Agent-Oriented Software Engineering, 1(3/4):370–395.

Malone, T. W. (1999).
Tools for inventing organizations: Toward a handbook of organizational process.
Management Science, 45(3):425–443.

Morin, E. (1977).
La méthode (1) : la nature de la nature.
Points Seuil.

Bibliography VI

Okuyama, F. Y., Bordini, R. H., and da Rocha Costa, A. C. (2008).
A distributed normative infrastructure for situated multi-agent organisations.
In Baldoni, M., Son, T. C., van Riemsdijk, M. B., and Winikoff, M., editors, *DALT*, volume 5397 of *Lecture Notes in Computer Science*, pages 29–46. Springer.

Ossowski, S. (1999).
Co-ordination in Artificial Agent Societies: Social Structures and Its Implications for Autonomous Problem-Solving Agents, volume 1535 of *LNAI*. Springer.

Piunti, M., Ricci, A., Boissier, O., and Hubner, J. (2009a).
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.

Piunti, M., Ricci, A., Boissier, O., and Hübner, J. F. (2009b).
Embodied organisations in mas environments.
In Braubach, L., van der Hoek, W., Petta, P., and Pokahr, A., editors, *Proceedings of 7th German conference on Multi-Agent System Technologies (MATES 09), Hamburg, Germany, September 9-11*, volume 5774 of *LNCS*, pages 115–127. Springer.

Bibliography VII

Pynadath, D. V. and Tambe, M. (2003).
An automated teamwork infrastructure for heterogeneous software agents and humans.
Autonomous Agents and Multi-Agent Systems, 7(1-2):71–100.

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.

Rocha Costa, A. C. d. and Dimuro, G. (2009).
A minimal dynamical organization model.
In Dignum, V., editor, *Multi-Agent Systems: Semantics and Dynamics of Organizational Models*, chapter XVII, pages 419–445. IGI Global.

Tambe, M. (1997).
Towards flexible teamwork.
Journal of Artificial Intelligence Reseearch, 7:83–124.