Programming organisational agents with MOISE\textsuperscript{+} & Jason

Jomi F. Hübner

ENS Mines Saint Etienne, France
hubner@emse.fr

Technical Fora Group at EUMAS’07
Software tools to build regulated MAS
Proposal

- Programming agents with a high abstraction level
  - AgentSpeak
  - BDI agents (reactive planning)
- Enable the programmer to state when the agent should adopt a role, a mission, ...
- Enable the agents to access organisational information
- Use Jason (open-source interpreter of AgentSpeak, developed by Rafael Bordini and Jomi Hübner)
by example: “writing a paper”

(a) Structural Specification

(b) Functional Specification

(c) Deontic Specification

editor  permission  mMan
writer  obligation  mCol
writer  obligation  mBib
General view
Example (AgentSpeak plan)

+some_event : some_context
  <- jmoise.create_group(wpgroup).

Some available Organisational Actions:

- For groups:
  - create_group(<GrSpecId>[,<GrId>])
  - remove_group(<GrId>)

- For schemes:
  - create_scheme(<SchSpecId> [, <responsible
groups>])
  - add_responsible_group(<SchId>,<GrId>)
  - remove_scheme(<SchId>)
  - set_goal_state(<SchId>,<Goal>,<State>)
Organisational Actions in AgentSpeak II

For Agents:
- `adopt_role(<RoleId>,<GrId>)`
- `remove_role(<RoleId>,<GrId>)`
- `commit_mission(<MisId>,<SchId>)`
- `remove_mission([<MisId>,] <SchId>)`

OrgManager will perform those actions in case they are consistent, e.g. the adoption of role is constrained by
- the cardinality of the role in the group
- the compatibilities of the roles played by the agent
Handling Organisational Events in AgentSpeak

Whenever something changes in the organisation, the organisation architecture updates the agent belief base accordingly.

Example (A new group is created)

\[+\text{group(wpgroup, GId)} : \text{true} \quad \text{<- jmoise.adopt_role(editor, GId).}\]

or

\[+\text{group(wpgroup, GId)[owner(O)] : my_friend(O)} \quad \text{<- jmoise.adopt_role(editor, GId).}\]

Example (Some group is destroyed)

\[-\text{group(wpgroup, GId)} \quad <\quad .\text{print("Group removed!")}.\]
Available Organisational Events I

- \(+/-\) `group(<GrSpecId>,<GrId>)[owner(<AgName>)]`: perceived by all agents when a group is created (event \(+\)) or removed (event \(-\)) by `AgName`.

- \(+/-\) `play(<AgName>,<RoleId>,<GrId>)`: perceived by the agents of `GrId` when an agent adopts (event \(+\)) or remove (event \(-\)) a role in group `GrId`.

- \(+/-\) `commitment(<AgName>,<MisId>,<SchId>)`: perceived by the `SchId` players when an agent commits or removes a commitment to a mission `MisId` in scheme `SchId`.
Available Organisational Events II

+/- scheme(<SchSpecId>,<SchId>)[owner(<AgName>)]:
  perceived by all agents when a scheme is created (+) or finished (-) by AgName.

+ scheme_group(<SchId>,<GrId>):
  perceived by GrId players when this group becomes responsible for the scheme SchId.

+ goal_state(<SchId>,<GoalId>,<State>):
  perceived by SchId players when the state of some goal changes.
Available Organisational Events III

+ **obligation**\( (< Schld >, < Misld >) \)

\[ \text{role}(< RoleId >), \text{group}(< Grld >) \]:

perceived by an agent when is has an organisational obligation for a mission. It has a role (\( RoleId \)) in a group (\( Grld \)) responsible for a scheme (\( Schld \)) and this role is obligated to a mission in this scheme.

+ **permission**\( (< Schld >, < Misld >) \)

\[ \text{role}(< RoleId >), \text{group}(< Grld >) \]:
Achieving Organisational Goals

An achievement goal event (+!g) is create when an organisational goal g is permitted.

Example (Organisational goal)

If an agent is committed to a mission with goal wsec, whenever this goal is possible (all its pre-condition goals are satisfied), the following plan may be selected:

+%wsec[scheme(Sch)]
  : commitment(A, mBib, Sch)
  <- ..... actions to write the section .....;
    .send(A,tell,[references]);
    jmoise.set_goal_state(Sch, wsec, satisfied).

The context of this plan uses organisational information to constraint its execution.
Execution sample I

jaime  action: jmoise.create_group(wpgroup)

all  perception: group(wpgroup,g1)[owner(jaime)]

jaime  action: jmoise.adopt_role(editor,g1)

olivier  action: jmoise.adopt_role(writer,g1)

jomi  action: jmoise.adopt_role(writer,g1)

all  perception:
  play(jaime,editor,g1)
  play(olivier,writer,g1)
  play(jomi,writer,g1)
Execution sample II

jaime  action: jmoise.create_scheme(writePaperSch, [g1])

all  perception: scheme(writePaperSch,s1)[owner(jaime)]

all  perception: scheme_group(s1,g1)

jaime  perception:
permission(s1,mManager)[role(editor),group(wpgroup)]

jaime  action: jmoise.commit_mission(mManager,s1)

olivier  perception:
obligation(s1,mColaborator)[role(writer),group(wpgroup),
obligation(s1,mBib)[role(writer),group(wpgroup)]

olivier  action: jmoise.commit_mission(mColaborator,s1)

olivier  action: jmoise.commit_mission(mBib,s1)
Execution sample III

jomi perception:
  obligation(s1,mColaborator)[role(writer),group(wpgroup),
  obligation(s1,mBib)[role(writer),group(wpgroup)]

jomi action: jmoise.commit_mission(mColaborator,s1)

all perception:
  commitment(jaime,mManager,s1)
  commitment(olivier,mColaborator,s1)
  commitment(olivier,mBib,s1)
  commitment(jomi,mColaborator,s1)
Execution sample IV

all  perception: goal_state(s1,*,unsatisfied)

jaime  goal: wtitle
action: jmoise.set_goal_state(s1,wtitle,satisfied)
(after each set_goal_state all agents have theirs beliefs updated)

jaime  goal: wabs
action: jmoise.set_goal_state(s1,wabs,satisfied)

jaime  goal: wsectitles
action: jmoise.set_goal_state(s1,wsectitles,satisfied)

jaime  goal: fdv
action: jmoise.set_goal_state(s1,fdv,satisfied)
Execution sample V

olivier  goal: wsecs
         action: jmoise.set_goal_state(s1,wsecs,satisfied)

jomi    goal: wsecs
         action: jmoise.set_goal_state(s1,wsecs,satisfied)

jaime   goal: wcon
         action: jmoise.set_goal_state(s1,wcon,satisfied)

olivier goal: wref
         action: jmoise.set_goal_state(s1,wref,satisfied)

olivier action: jmoise.set_goal_state(s1,sv,satisfied)

jaime   goal: wpGoal
         action: jmoise.set_goal_state(s1,wpGoal,satisfied)
Execution sample VI

all  action: jmoise.remove_mission(s1)

jaime  action: jmoise.jmoise.remove_scheme(s1)
Demo

- Agents’ sources
- Application Execution
- Debugging
Summary

- A tool to program $\text{MOISE}^+$ agents
  - Logic
  - BDI
  - AgentSpeak

- $\text{J-MOISE}^+$
  - OrgManager
  - Organisational actions
  - Organisational events

- An implementation is available at http://jason.sourceforge.net
