Detailed description of ORA4MAS demo

Rosine Kitio[1], Michele Piunti[2], Jomi Fred Hübner[1]*, and Olivier Boissier[1]

[1] SMA/G2I/ENSM.SE, 158 Cours Fauriel
42023 Saint-Etienne Cedex, France
{boissier,hubner,kitio}@emse.fr

[2] DEIS, ALMA MATER STUDIORUM Università di Bologna
47023 Cesena (FC), Italy
m.piunti@unibo.it

1 Introduction

As described in the paper submitted with our demonstration, ORA4MAS is an infrastructure to support the management of open multi-agent organisations where agents can enter/exit the organisation at anytime, cooperate with each other to achieve goals with compliant or non compliant behaviors. For running the demo and the system, it is required to use (i) the java development kit version 1.6 (jdk 1.6) and the ant tool. In the following sections, we present step by step, with a simple example, the functionning of the system from its deployment to the creation of the organisational artifacts and the agents' interactions with these artifacts.

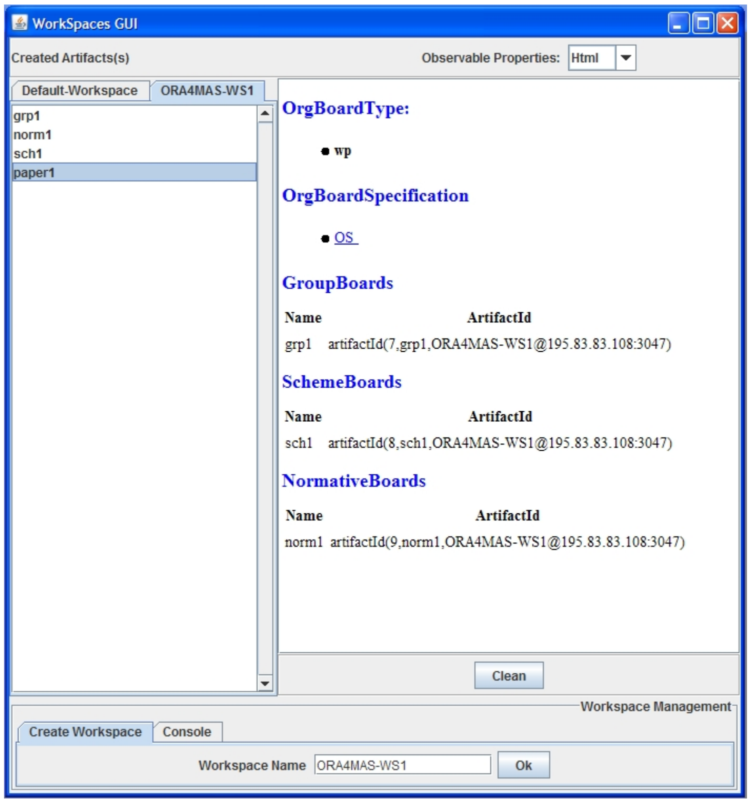
2 Running example writing paper

The axample we present in this demo is an organisation designed to allow agents to write a paper together.

The cooperation pattern aiming at helping and coordinating their behaviours is available as an organisational specification (file `wpOS.xml` in the folder of the demo) written with the $\mathcal{M}$OISE$^+$ Organisation Modeling Language. The organisation specification (OS) is composed of one group `wpgroup` composed of two roles: *writer* and *editor*. The functional specification of the OS named `writePaperSch` defines three missions that the agents can commit to: *mManager*, *mBib*, and *mCollaborator*. A subset of goals are related to each mission so that, an agent which commit to a mission is supposed to achieve these goals. The deontic specification which links the structural specification and the functional specification states the norms (obligation, permission, prohibition) between the roles and the missions. Instances of norms are assigned to the agents according to the roles that they adopt. All behaviour which is not explicitly permitted or obliged by a norm is considered as a prohibition. However an agent, being autonomous, can decide to be not compliant to a norm.

For our scenario, we consider four agents (*Tom*, *Bob*, *John* and *Alice*) as co-authors of a paper. The agent *John* has been programmed in the Jadex Agent Programming Language, while the agents *Bob*, *Tom* and *Alice* have been programmed with the Jason Agent Programming Language. In this example, the agent *Tom* plays the role *editor* while the other agents play the role *writer*. The demonstration aims at presenting how this organisation composed of four agents is supported by ORA4MAS: creation, evolution, achievement of goals, ... In the following, we give the instructions for running and using the demo which is available at: http://moise.sourceforge.net/demos/ora4mas/. We also make some comments of the different results of the execution of each step. We chose to implement an agent in Jadex and others in Jason, to illustrate the ability of our infrastructure of managing the participation of heterogeneous agents in the same organisation.

* Supported by ANR Project ForTrust.



OrgBoardType:

- wp

OrgBoardSpecification

- OS

GroupBoards

Name	ArtifactId
grp1	artifactId(7,grp1,ORA4MAS-WS1@195.83.83.108:3047)

SchemeBoards

Name	ArtifactId
sch1	artifactId(8,sch1,ORA4MAS-WS1@195.83.83.108:3047)

NormativeBoards

Name	ArtifactId
norm1	artifactId(9,norm1,ORA4MAS-WS1@195.83.83.108:3047)

Fig. 1. ORA4MAS Demo: Screen1 - center: values of the observable property of the OrgBoard **paper1** after Step1

2.1 Step0: Deploiement of ORA4MAS

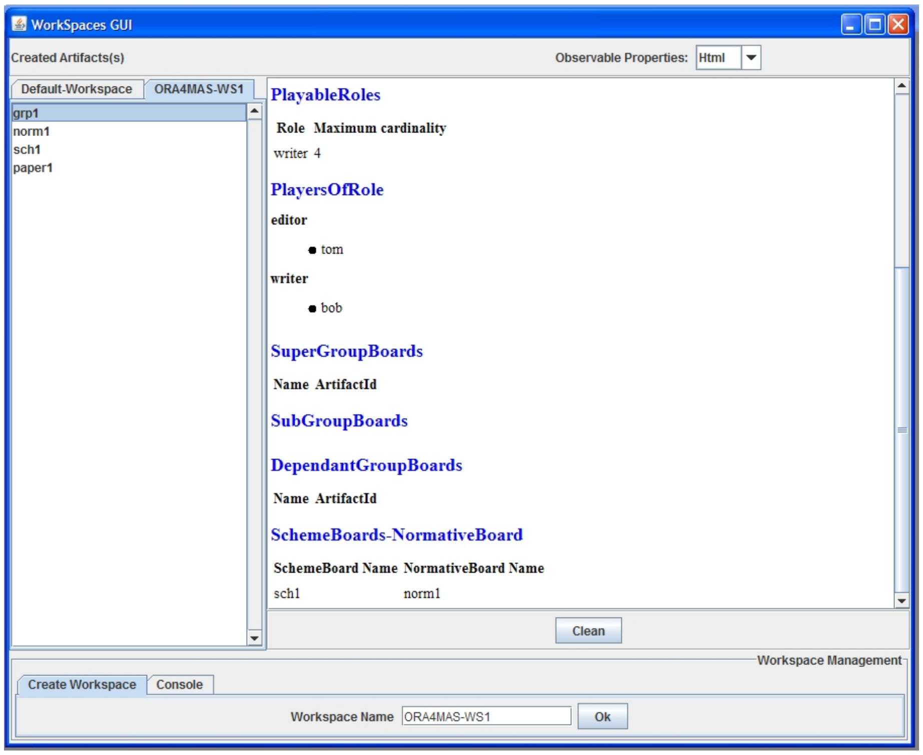
The ORA4MAS infrastructure is launched by the following command shell:

```
ant ora4mas.
```

This command launches the *Cartago* middleware. It creates the graphical user interface (GUI) where are shown the initial workspace "*Default-Workspace*" This GUI is divided in three main parts (see Fig. 1):

left panel : all created workspace(s) are shown with their corresponding artifacts.
center panel : current value/state of the observable properties of the selected artifact on the left panel.
bottom panel : interaction field for the user to create new workspaces.

To deploy the ORA4MAS infrastructure supporting the organisation used in the demonstration, we create a new workspace "*ORA4MAS-WS1*" from the GUI window.



Workspace Name: ORA4MAS-WS1

PlayableRoles

Role	Maximum cardinality
writer	4

PlayersOfRole

editor

- tom

writer

- bob

SuperGroupBoards

Name	ArtifactId

SubGroupBoards

DependantGroupBoards

Name	ArtifactId

SchemeBoards-NormativeBoard

SchemeBoard Name	NormativeBoard Name
sch1	norm1

Fig. 2. ORA4MAS Demo: Screen2 - Value of the observable property of the GroupBoard grp1 after Step1

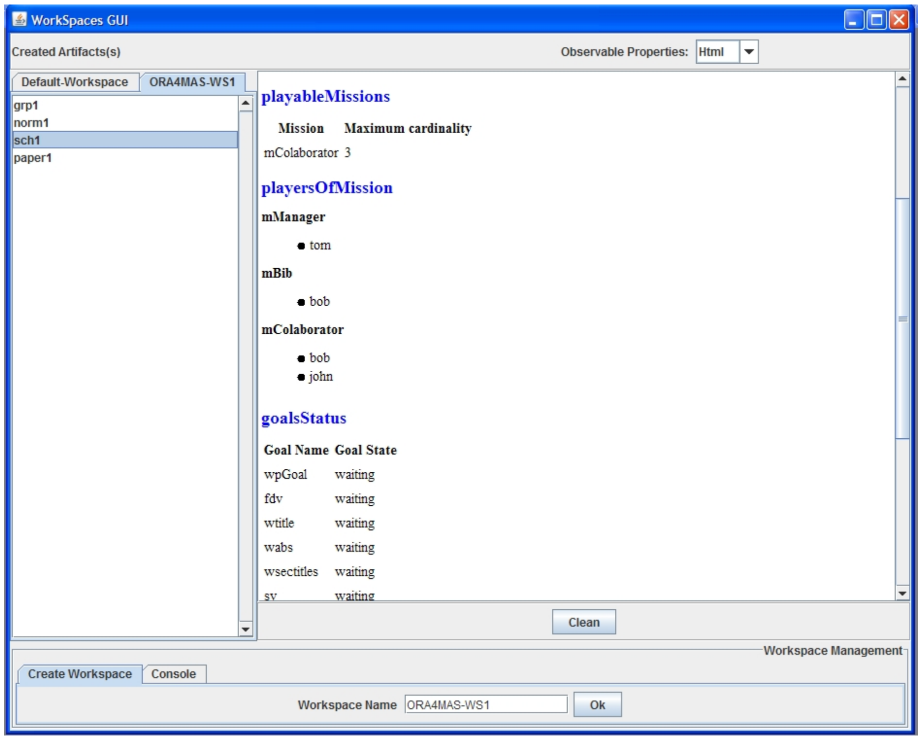
2.2 Step1: Jason Agents enter the organisation

In the command shell, agents *Tom* and *Bob* are launched (both are programmed in Jason).

ant step1.

The agent *Tom* has the possibility to take decisions with respect to the organisation life cycle. Therefore, at the beginning of its execution, it deploys the organisational artifacts supporting the organisation which is specified in `wpOS.xml`: OrgBoard *paper1*, followed by GroupBoard *grp1*, SchemeBoard *sch1* and NormativeBoard *norm1*. Those artifacts can be seen in the left panel of the ORA4MAS window by exploring the workspace "*ORA4MAS-WS1*". By selecting the organisational artifact *paper1*, the value of its observable properties are displayed on the center panel (cf. Fig. 1). They are essentially the list of the created GroupBoards, SchemeBoards and NormativeBoards in this organisation.

After having created the organisational artifacts, agent *Tom* adopts the role *editor* and commits to the mission *mManager* while agent *Bob* adopts the role *writer* and commits to the missions *mBib* and *mColaborator*. We can see the result of these actions by displaying observable properties of the GroupBoard `grp1` (cf. Fig. 2). Before continuing the demo execution, we can also explore the current values of the observable properties of the other organisational artifacts by selecting in the GUI window.



grp1
norm1
sch1
paper1

playableMissions

Mission	Maximum cardinality
mColaborator	3

playersOfMission

mManager

- tom

mBib

- bob

mColaborator

- bob
- john

goalsStatus

Goal Name	Goal State
wpGoal	waiting
fdv	waiting
wtitle	waiting
wabs	waiting
wsectitles	waiting
sv	waiting

Fig. 3. ORA4MAS Demo: Screen3 - on the center value of the observable properties of the SchemeBoard **sch1** after Step3

2.3 Step2: Jadex agent enters the organisation

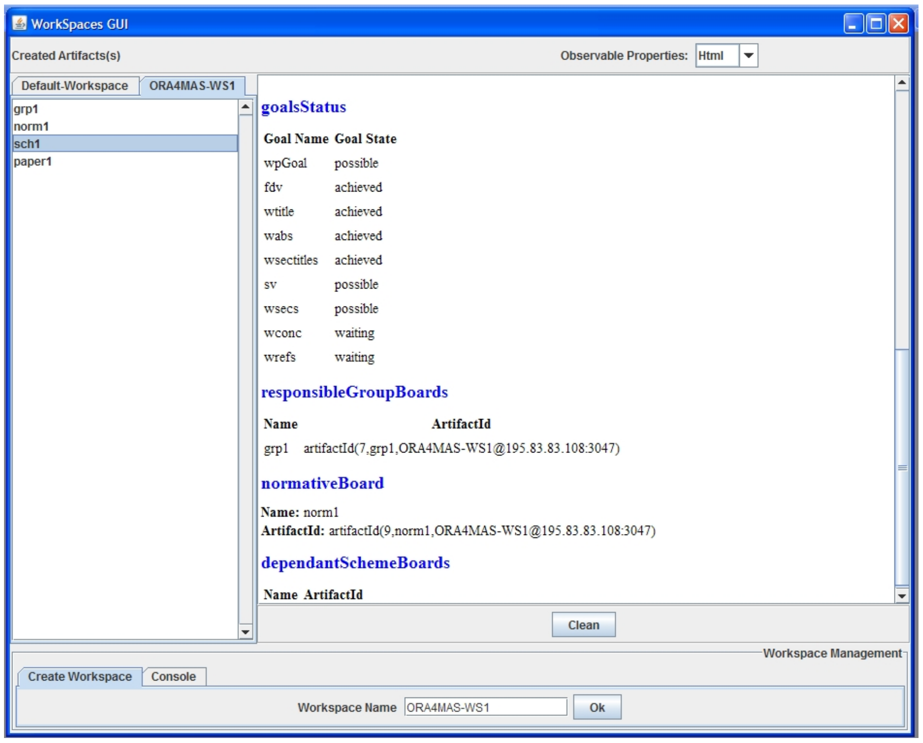
The agent *John* is launched by the way of the GUI of Jadex platform that is deployed with the following shell command:

ant step2.

On the GUI of Jadex we need to set the path where is registered the agent source. For our example this path is Jadex/src/exampleWritingPaper/writerAgent.xml. We can also change the name of the agent in the corresponding location on the Jadex GUI and set the name *John*. As soon as created, agent *John* adopts the role *writer* and commits to the mission *mColaborator*.

Using the GUI window, we can observe the properties of the organisational artifacts. Let's notice that the maximum cardinality[3] of the property `playableMissions` of the SchemeBoard *sch1* is now 3 and the state of all the goals is still `waiting`(cf. Fig. 4). This is because, the property minimum cardinality of agent(s) for each mission in the functional specification define the constraint that, the goals of a social

[3] The initial value of the cardinality of each mission is defined in the functional specification by the designer. We can display the specification of an organisational artifact by clicking on the corresponding observable property (OS, SS, FS, NS) on the center panel.



goalsStatus

Goal Name	Goal State
wpGoal	possible
fdv	achieved
wtitle	achieved
wabs	achieved
wsectitles	achieved
sv	possible
wsecs	possible
wconc	waiting
wrefs	waiting

responsibleGroupBoards

Name	ArtifactId
grp1	artifactId(7,grp1,ORA4MAS-WS1@195.83.83.108:3047)

normativeBoard

Name: norm1
ArtifactId: artifactId(9,norm1,ORA4MAS-WS1@195.83.83.108:3047)

dependantSchemeBoards

Name	ArtifactId

Fig. 4. ORA4MAS Demo: Screen4 - on the center value of the observable properties of the SchemeBoard **sch1** after Step4

scheme can not be achieved if the value of the minimum cardinality for each mission is not reached. In our example one agent is still needed for the mission *mColaborator* since its minimum cardinality is three.

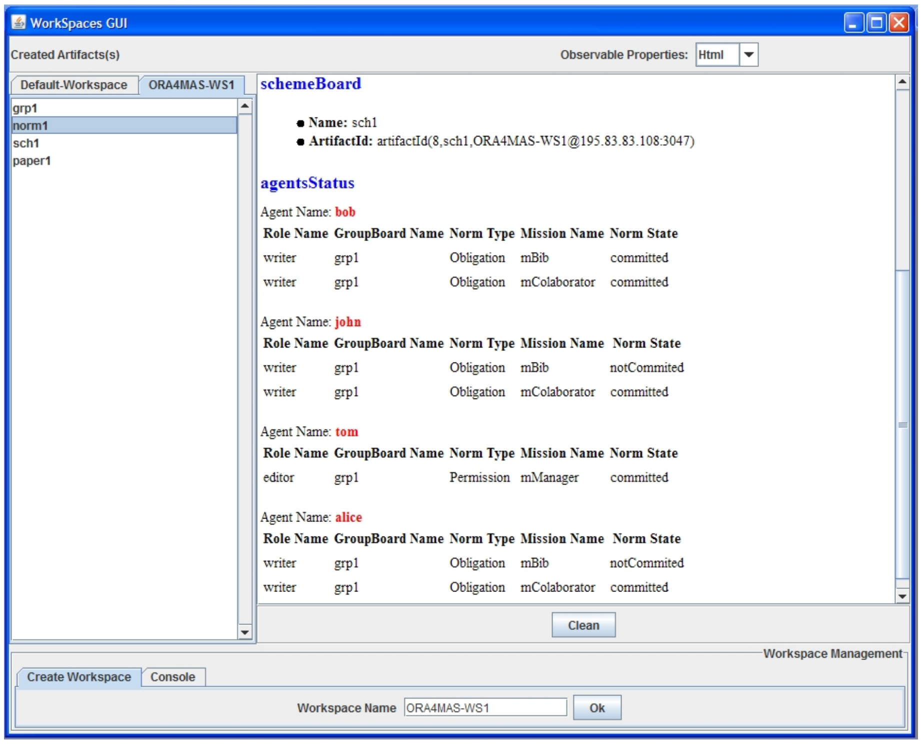
2.4 Step3: Alice enters the organisation

The last agent *Alice* is launched with the command shell:

```
ant step3
```

It enters the organisation. The minimum of agents being reached, the execution of the different goals to write the paper as specified in the social scheme of the FS started. The result of this step can be seen in the GUI window by inspecting the content of the different artifacts as explained above.

– In Fig. 4, we can see that the value of some goals have changed to `achieved` while others are `possible`.
– In the NormativeBoard *norm1* the observable property `agentStatus` has also been updated with the agent *Alice*. We can also see on Fig. 5 the list of the agents of the organisation and their related norms with the current state of each of these norms.



Observable Properties: Html

schemeBoard

- **Name:** sch1
- **ArtifactId:** artifactId(8,sch1,ORA4MAS-WS1@195.83.83.108:3047)

agentsStatus

Agent Name: **bob**

Role Name	GroupBoard Name	Norm Type	Mission Name	Norm State
writer	grp1	Obligation	mBib	committed
writer	grp1	Obligation	mColaborator	committed

Agent Name: **john**

Role Name	GroupBoard Name	Norm Type	Mission Name	Norm State
writer	grp1	Obligation	mBib	notCommited
writer	grp1	Obligation	mColaborator	committed

Agent Name: **tom**

Role Name	GroupBoard Name	Norm Type	Mission Name	Norm State
editor	grp1	Permission	mManager	committed

Agent Name: **alice**

Role Name	GroupBoard Name	Norm Type	Mission Name	Norm State
writer	grp1	Obligation	mBib	notCommited
writer	grp1	Obligation	mColaborator	committed

Workspace Name ORA4MAS-WS1

Fig. 5. ORA4MAS Demo: Screen4 - Value of the observable properties of the NormativeBoard **norm1** after Step4

3 Conclusion

Here we have taken a simple example with only one GroupBoard, one SchemeBoard, one NormativeBoard just to present the main features and functionalities of ORA4MAS infrastructure. We chose to keep this example for the expected live demo presentation in order to avoid a very long demo description in this paper. We can however suitably run an organisation with several artifacts which are independent or structured in an hierarchy (in case of an organisation with group and sub-groups as can be stated in $\mathcal{M}$OISE$^+$). For instance, we also tested our implementation with a soccer team with two sub groups `defense` and `attack` belonging in one main `team` where the agents of each of the group cooperate to achieve their common objective to *score a goal* with a defined social scheme. We plan to show also this implementation during the demo session at AAMAS in case our proposal is accepted.