Flexible hierarchical organisation of role based agents

Emmanuel Adam   Emmanuelle Grislin-Le Strugeon   Rene Mandiau

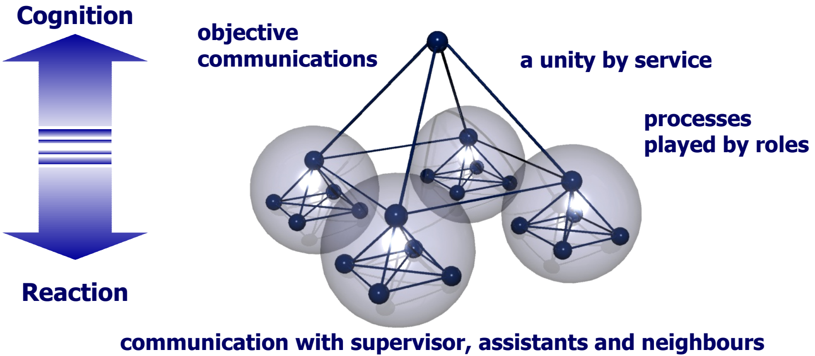
LAMIH (UMR CNRS 8530)
Université de Valenciennes, FRANCE

SARC'08

1. Background : bring MAS into human organization
2. Proposition of a holomas using roles
   - Elements of formalisation
   - Architecture
3. Dynamics in holonic multiagent organisation
   - Dynamic of roles
   - Robustness
   - Growth
4. Implementation
5. Application
6. Perspectives and Conclusion
   - Perspectives
   - Conclusion

Background : bring MAS into human organization

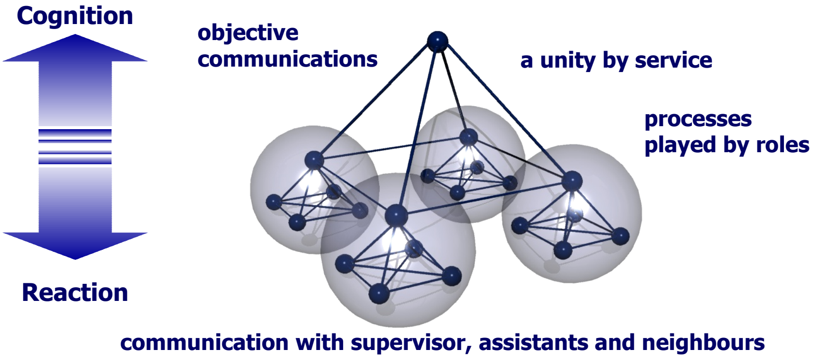
Application/Social Context



Background : bring MAS into human organization

Application/Social Context

- Integration of multi-agent organization into human organization



Cognition

Reaction

objective communications

a unity by service

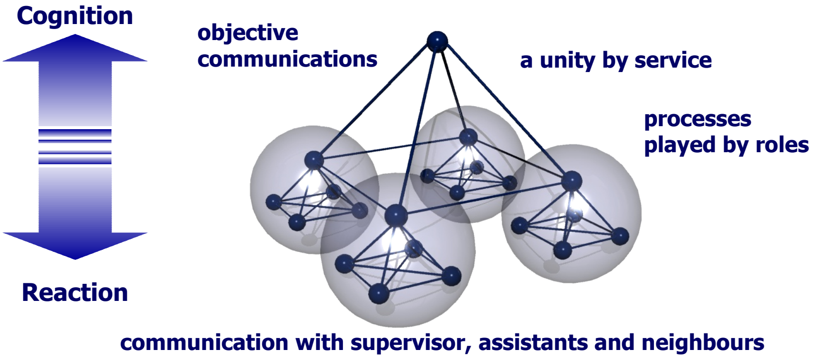
processes played by roles

communication with supervisor, assistants and neighbours

Background : bring MAS into human organization

Application/Social Context

- Integration of multi-agent organization into human organization
- Administrative systems as Holonic organizations



Cognition

Reaction

objective communications

a unity by service

processes played by roles

communication with supervisor, assistants and neighbours

Proposition of a holomas using roles

Roles

Proposition of a holomas using roles

Roles

- According to the propositions in [Koestler 69] (canon,rules,skills, strategies), a role is composed of a set of rules :

Proposition of a holomas using roles

Roles

- According to the propositions in [Koestler 69] (canon,rules,skills, strategies), a role is composed of a set of rules :

  mechanism.1 : "Functional holons are governed by fixed sets of rules and display more or less flexible strategies."

Proposition of a holomas using roles

Roles

- According to the propositions in [Koestler 69] (canon,rules,skills, strategies), a role is composed of a set of rules :

  mechanism.1 : "Functional holons are governed by fixed sets of rules and display more or less flexible strategies."

  mechanism.2 : "The rules, referred to as the system's canon, determine its invariant properties, its structural configuration and/or functional pattern."

Proposition of a holomas using roles

Roles

- According to the propositions in [Koestler 69] (canon,rules,skills, strategies), a role is composed of a set of rules :

mechanism.1 : "Functional holons are governed by fixed sets of rules and display more or less flexible strategies."

mechanism.2 : "The rules, referred to as the system's canon, determine its invariant properties, its structural configuration and/or functional pattern."

mechanism.3 : "While the canon defines the permissible steps in the holon's activity, the strategic selection of the actual step among permissible choices is guided by the contingencies of the environment."

Proposition of a holomas using roles

Roles

- According to the propositions in [Koestler 69] (canon,rules,skills, strategies), a role is composed of a set of rules :

mechanism.1 : "Functional holons are governed by fixed sets of rules and display more or less flexible strategies."

mechanism.2 : "The rules, referred to as the system's canon, determine its invariant properties, its structural configuration and/or functional pattern."

mechanism.3 : "While the canon defines the permissible steps in the holon's activity, the strategic selection of the actual step among permissible choices is guided by the contingencies of the environment."

mechanism.4 : "Holons on successively higher levels of the hierarchy show increasingly complex, more flexible and less predictable patterns of activity, while on successive lower levels we find increasingly mechanised, stereotyped and predictable pattern."

Rules formalisation

Rules

- we define a rule as a set of behaviours :

$$R = (name_R, priority_R, tasks_R)$$

$$tasks_R = \left\{ t_0^R, \ldots, t_{nt}^R \right\}$$

$$nt = number\ of\ tasks$$

Roles formalisation

Roles

- We define a Role as a set of essential rules and a set of secondary rules
- *A searcher has to publish (a lot of) articles. To help the laboratory, he/she can manage the library, the projects, the phd students . . . . . .*

$$role = \left(\begin{array}{l} name, priority, KP, KE, KS, \\ hardRules, flexibleRules \end{array} \right)$$

KP : Pre-requirement, consequences, weight,
KE : Environmental Knowledge (data)
KS : Social Knowledge (roles names and constraints)

- *For example : a speaker has to respect the time-limit fixed by the chair-man*

Agents formalisation

Agents

Holonic Agents : Our holonic agents are defined as follow :

$$agent_a = \begin{pmatrix} KP, KE, KS, HRA, messages_a, \\ perception_a, rules_a, roles_a \end{pmatrix}$$

KP : $\textit{(Personal knowledge)} = \{name; current\ \ state; individual\ \ goals(GI)\}$

KE : *(Environmental knowledge)* = partial representations of objects of the environment.

KS : *(Social knowledge)* = partial representations of the acquaintances & collective goal (GC).

Agents formalisation

Agents

Holonic Agents : Our holonic agents are defined as follow :

$$agent_a = \begin{pmatrix} KP, KE, KS, HRA, messages_a, \\ perception_a, rules_a, roles_a \end{pmatrix}$$

KP : *(Personal knowledge)* $= \{name; current\ state; individual\ goals(GI)\}$

KE : *(Environmental knowledge)* = partial representations of objects of the environment.

KS : *(Social knowledge)* = partial representations of the acquaintances & collective goal (GC).

HRA : *(Holonic Roles Agent)* = Agent that manages roles of the system.

MAS formalisation

MAS

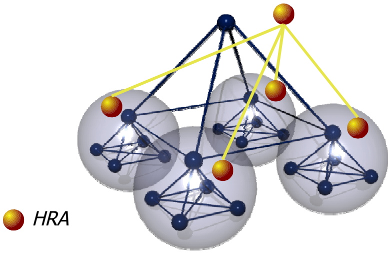
Mas and agent definition : MAS is simply defined as a set of agents.

Environment definition : $E = \{object_0, object_1, ..., object_n\}$

World definition : $world = (environment, mas)$

General architecture of our Holonic IMAS

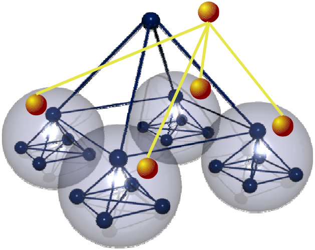
Architecture



General architecture of our Holonic IMAS

Architecture

- Each of our Holomas is assisted by a *HRA* that manages the roles

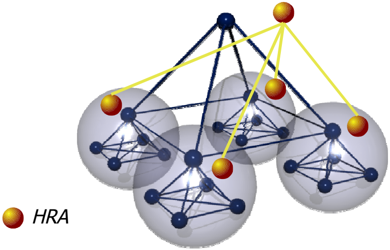


● *HRA*

General architecture of our Holonic IMAS

Architecture

- Each of our Holomas is assisted by a *HRA* that manages the roles
- A HRA could be a set of HRA distributed around the Holomas



Dynamic of roles

Dynamic of roles

Dynamic of roles

Dynamic of roles

- Priority of a role's rule increases each time the agent chooses it.

Dynamic of roles

Dynamic of roles

- Priority of a role's rule increases each time the agent chooses it.
- Each agent has its definition of a role.

Dynamic of roles

Dynamic of roles

- Priority of a role's rule increases each time the agent chooses it.
- Each agent has its definition of a role.
- Each HRA agent checks if there are differences between prescribed roles and roles really played by agents.

Dynamic of roles

Dynamic of roles

- Priority of a role's rule increases each time the agent chooses it.
- Each agent has its definition of a role.
- Each HRA agent checks if there are differences between prescribed roles and roles really played by agents.
- New definition of a role is kept if a sufficient number of agents have modified a role in the same way.

Dynamic of roles

Dynamic of roles

- Priority of a role's rule increases each time the agent chooses it.
- Each agent has its definition of a role.
- Each HRA agent checks if there are differences between prescribed roles and roles really played by agents.
- New definition of a role is kept if a sufficient number of agents have modified a role in the same way.
- A secondary rule can become a hard rule if all agents always choose it.

Robustness by replication in holonic multi-agent organisation

Robustness

Robustness by replication in holonic multi-agent organisation

Robustness

- Two critical agents : responsible of the Holonic Roles Agents & the Holomas responsible agent.

Robustness by replication in holonic multi-agent organisation

Robustness

- Two critical agents : responsible of the Holonic Roles Agents & the Holomas responsible agent.
    - physical replication, ping messages

Robustness by replication in holonic multi-agent organisation

Robustness

- Two critical agents : responsible of the Holonic Roles Agents & the Holomas responsible agent.
  - physical replication, ping messages
- For other agents, a watch holon-assistants, allows to detect damaged agents

Robustness by replication in holonic multi-agent organisation

Robustness

- Two critical agents : responsible of the Holonic Roles Agents & the Holomas responsible agent.
  - physical replication, ping messages
- For other agents, a watch holon-assistants, allows to detect damaged agents
  - creation of a new agent by the responsible of the faulty agent

Robustness by replication in holonic multi-agent organisation

Robustness

- Two critical agents : responsible of the Holonic Roles Agents & the Holomas responsible agent.
  - physical replication, ping messages
- For other agents, a watch holon-assistants, allows to detect damaged agents
  - creation of a new agent by the responsible of the faulty agent
  - re-load of roles and links from the HRA

Robustness by replication in holonic multi-agent organisation

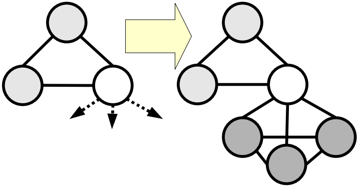
Robustness

- Two critical agents : responsible of the Holonic Roles Agents & the Holomas responsible agent.
    - physical replication, ping messages
- For other agents, a watch holon-assistants, allows to detect damaged agents
    - creation of a new agent by the responsible of the faulty agent
    - re-load of roles and links from the HRA
    - problem : loss of the data used during the breakdown

Growth in holonic multi-agent organisation

Growth when overload

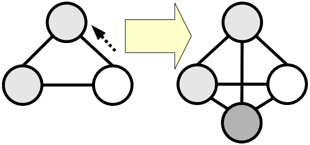
- Downward Growth : The agent creates assistants and delegates some of its tasks to them.



Growth in holonic multi-agent organisation

Growth when overload

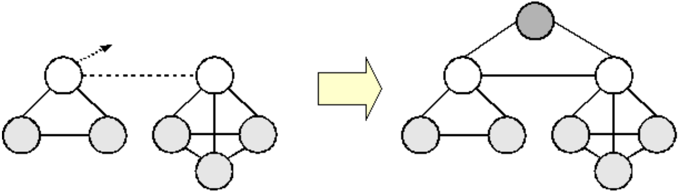
- Downward Growth : The agent creates assistants and delegates some of its tasks to them.
- Horizontal growth : an agent needs a skill and its responsible creates a neighbour.



Growth in holonic multi-agent organisation

Growth when overload

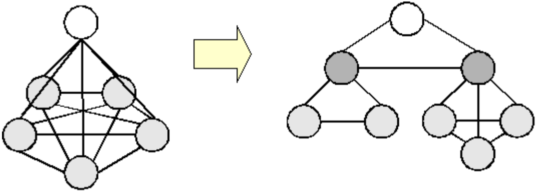
- Downward Growth : The agent creates assistants and delegates some of its tasks to them.
- Horizontal growth : an agent needs a skill and its responsible creates a neighbour.
- Upward Growth : To coordinate agents / systems acting towards a same goal.



Growth in holonic multi-agent organisation

Growth when overload

- Downward Growth : The agent creates assistants and delegates some of its tasks to them.
- Horizontal growth : an agent needs a skill and its responsible creates a neighbour.
- Upward Growth : To coordinate agents / systems acting towards a same goal.
- Internal Growth : A HoloMAS agent creates internal coordinators.



Well-balanced growth

Well-balanced growth

Well-balanced growth

Well-balanced growth

- Needs :

Well-balanced growth

Well-balanced growth

- Needs :
    - Downward and horizontal growth must respect holonic concepts : an assistant deal with less complex roles than its responsible

Well-balanced growth

Well-balanced growth

- Needs :
  - Downward and horizontal growth must respect holonic concepts : an assistant deal with less complex roles than its responsible
  - Growth must produce a well-balanced holarchy

Well-balanced growth

Well-balanced growth

- Needs :
  - Downward and horizontal growth must respect holonic concepts : an assistant deal with less complex roles than its responsible
  - Growth must produce a well-balanced holarchy
- Use of :

Well-balanced growth

Well-balanced growth

- Needs :
  - Downward and horizontal growth must respect holonic concepts : an assistant deal with less complex roles than its responsible
  - Growth must produce a well-balanced holarchy
- Use of :
  - agent workload (wl)

Well-balanced growth

Well-balanced growth

- Needs :
  - Downward and horizontal growth must respect holonic concepts : an assistant deal with less complex roles than its responsible
  - Growth must produce a well-balanced holarchy
- Use of :
  - agent workload (*wl*)
  - agents maximum workload thresholds (*mwt*)

Well-balanced growth

Well-balanced growth

- Needs :
  - Downward and horizontal growth must respect holonic concepts : an assistant deal with less complex roles than its responsible
  - Growth must produce a well-balanced holarchy
- Use of :
  - agent workload (*wl*)
  - agents maximum workload thresholds (*mwt*)
  - workload of a group (sub-holomas) (*gwl*)

Well-balanced growth

Well-balanced growth

- Needs :
  - Downward and horizontal growth must respect holonic concepts : an assistant deal with less complex roles than its responsible
  - Growth must produce a well-balanced holarchy
- Use of :
  - agent workload (*wl*)
  - agents maximum workload thresholds (*mwt*)
  - workload of a group (sub-holomas) (*gwl*)
  - agent workload still available (*wla*)

Well-balanced growth

Well-balanced growth

- Needs :
  - Downward and horizontal growth must respect holonic concepts : an assistant deal with less complex roles than its responsible
  - Growth must produce a well-balanced holarchy
- Use of :
  - agent workload (*wl*)
  - agents maximum workload thresholds (*mwt*)
  - workload of a group (sub-holomas) (*gwl*)
  - agent workload still available (*wla*)
- Note : Use of the ContractNet protocol

Well-balanced growth

Well-balanced growth

- Needs :
  - Downward and horizontal growth must respect holonic concepts : an assistant deal with less complex roles than its responsible
  - Growth must produce a well-balanced holarchy
- Use of :
  - agent workload (wl)
  - agents maximum workload thresholds (mwt)
  - workload of a group (sub-holomas) (gwl)
  - agent workload still available (wla)
- Note : Use of the ContractNet protocol
- Note : if a_1 is the assistant of a_0, $mwt_{a_1} = \alpha \times mwt_{a_0}$ with $(\alpha \in]0, 1[)$

Well-balanced growth

Well-balanced growth

```
procedure HANDLECFP($ACLMessage\ cfp$)
    $loadAsked \leftarrow cfp.getContent()$
    $maxAssistantLoad \leftarrow \alpha \times mwt$
    $wla \leftarrow mwt - holonCurrentLoad$
    if $(wla - loadAsked) >= 0 \vee ((maxAssistantLoad - loadAsked) >= 0 \wedge \neg holon.isLeaf()))$ then
        if $(wla - loadAsked) > 0$ then
            RETURN($wla, gwl$)
        else
            RETURN($maxAssistantLoad, gwl$)
        end if
    else
        RETURN(Refuse)
    end if
end procedure
```

Implementation of our architecture

Implementation

Implementation of our architecture

Implementation

- Use of JADE platform, based on empty agents that receive behaviours

Implementation of our architecture

Implementation

- Use of JADE platform, based on empty agents that receive behaviours
- Role currently implemented as a set of behaviours (not a set of hard and flexible rules)

Implementation of our architecture

Implementation

- Use of JADE platform, based on empty agents that receive behaviours
- Role currently implemented as a set of behaviours (not a set of hard and flexible rules)
- core behaviours used by agents :

Implementation of our architecture

Implementation

- Use of JADE platform, based on empty agents that receive behaviours
- Role currently implemented as a set of behaviours (not a set of hard and flexible rules)
- core behaviours used by agents :

HolonicMessagesManagement : owned by all agents ; it allows an agent to handle message relative to : the acquaintances updating ; the role transmission ; the growth ;

Implementation of our architecture

Implementation

- Use of JADE platform, based on empty agents that receive behaviours
- Role currently implemented as a set of behaviours (not a set of hard and flexible rules)
- core behaviours used by agents :

HolonicMessagesManagement : owned by all agents ; it allows an agent to handle message relative to : the acquaintances updating ; the role transmission ; the growth ;

RegenerationManagement : allows an agent to be survey by its acquaintances

Implementation of our architecture

Implementation

- Use of JADE platform, based on empty agents that receive behaviours
- Role currently implemented as a set of behaviours (not a set of hard and flexible rules)
- core behaviours used by agents :

HolonicMessagesManagement : owned by all agents ; it allows an agent to handle message relative to : the acquaintances updating ; the role transmission ; the growth ;

RegenerationManagement : allows an agent to be survey by its acquaintances

RoleManagement : behaviour is linked the roles manager agents that could be compared to Directory Facilitator agents

Implementation of our architecture

Implementation

- Use of JADE platform, based on empty agents that receive behaviours
- Role currently implemented as a set of behaviours (not a set of hard and flexible rules)
- core behaviours used by agents :

HolonicMessagesManagement : owned by all agents ; it allows an agent to handle message relative to : the acquaintances updating ; the role transmission ; the growth ;

RegenerationManagement : allows an agent to be survey by its acquaintances

RoleManagement : behaviour is linked the roles manager agents that could be compared to Directory Facilitator agents

ContractNetDelegation : linked temporarily to the initiator of the ContractNet protocol relative to the delegation of roles

Implementation of our architecture

Implementation

- Use of JADE platform, based on empty agents that receive behaviours
- Role currently implemented as a set of behaviours (not a set of hard and flexible rules)
- core behaviours used by agents :

HolonicMessagesManagement : owned by all agents ; it allows an agent to handle message relative to : the acquaintances updating ; the role transmission ; the growth ;

RegenerationManagement : allows an agent to be survey by its acquaintances

RoleManagement : behaviour is linked the roles manager agents that could be compared to Directory Facilitator agents

ContractNetDelegation : linked temporarily to the initiator of the ContractNet protocol relative to the delegation of roles

ContractNetService : linked temporarily to the potential responders of the ContractNet protocol.

Well balanced growth : Example 1

Example 1 of well-balanced growth

Well balanced growth : Example 1

Example 1 of well-balanced growth

- three initial agents : a_1 having a *mwt* of 10, responsible for $a_{1.1}$ and $a_{1.2}$

Well balanced growth : Example 1

Example 1 of well-balanced growth

- three initial agents : a_1 having a *mwt* of 10, responsible for $a_{1.1}$ and $a_{1.2}$
- three roles :$HeavyRole(HR)$, $load = 8$; $MediumRole(MR)$, $load = 5$ and $LightRole(LR)$, $load = 2$

Well balanced growth : Example 1

Example 1 of well-balanced growth

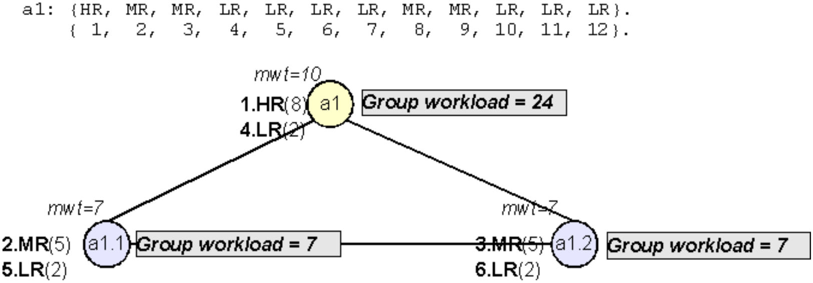
- three initial agents : a_1 having a *mwt* of 10, responsible for $a_{1.1}$ and $a_{1.2}$
- three roles :$HeavyRole(HR)$, $load = 8$; $MediumRole(MR)$, $load = 5$ and $LightRole(LR)$, $load = 2$
- $\alpha = 0.75 \Rightarrow mwt_{a_{1.1}} = mwt_{a_{1.2}} = 7$

Well balanced growth : Example 1

Example 1 of well-balanced growth

- three initial agents : a_1 having a *mwt* of 10, responsible for $a_{1.1}$ and $a_{1.2}$
- three roles :$HeavyRole(HR)$, $load = 8$; $MediumRole(MR)$, $load = 5$ and $LightRole(LR)$, $load = 2$
- $\alpha = 0.75 \Rightarrow mwt_{a_{1.1}} = mwt_{a_{1.2}} = 7$
- scenario = add the roles $\{HR, MR, MR, LR, LR, LR, LR, MR, MR, LR, LR, LR\}$ to a_1

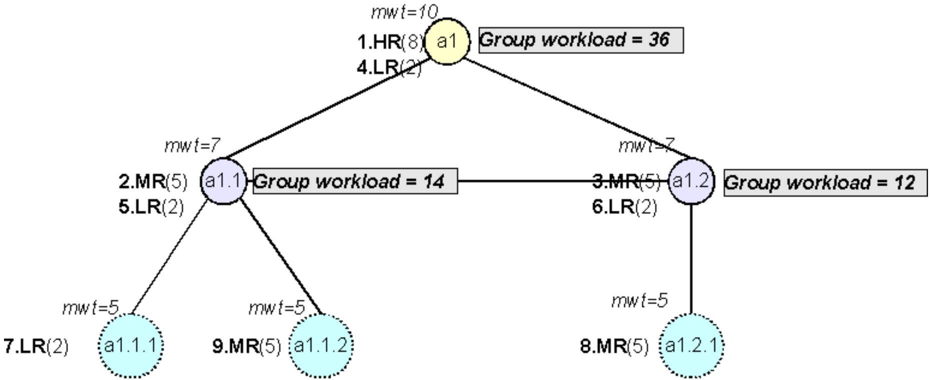
Well balanced growth : Example 1



```
a1: {HR, MR, MR, LR, LR, LR, LR, MR, MR, LR, LR, LR}.
    { 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12}.
```

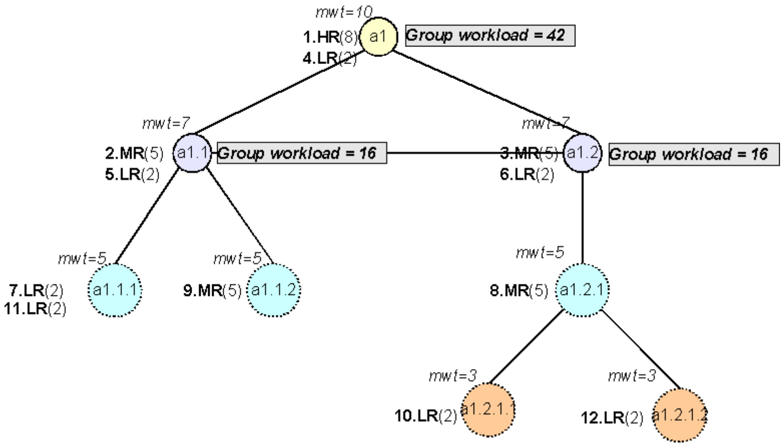
Well balanced growth : Example 1

```
a1: {HR, MR, MR, LR, LR, LR, LR, MR, MR, LR, LR, LR}.
    { 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12}.
```



Well balanced growth : Example 1

```
a1: {HR, MR, MR, LR, LR, LR, LR, MR, MR, LR, LR, LR}.
    { 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12}.
```



Well balanced growth - Example 2

Example 2 of well-balanced growth

Well balanced growth - Example 2

Example 2 of well-balanced growth

- same three initial agents a_1, $a_{1.1}$ and $a_{1.2}$; but with $mwt_{a_1} = 8$

Well balanced growth - Example 2

Example 2 of well-balanced growth

- same three initial agents a_1, $a_{1.1}$ and $a_{1.2}$; but with $mwt_{a_1} = 8$
- same three roles :$HeavyRole(HR)$, $load = 8$; $MediumRole(MR)$, $load = 5$ and $LightRole(LR)$, $load = 2$

Well balanced growth - Example 2

Example 2 of well-balanced growth

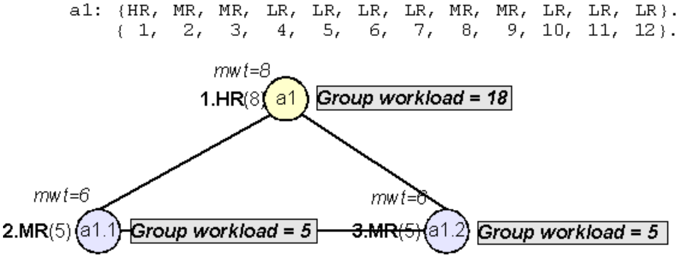
- same three initial agents a_1, $a_{1.1}$ and $a_{1.2}$; but with $mwt_{a_1} = 8$
- same three roles :$HeavyRole(HR)$, $load = 8$; $MediumRole(MR)$, $load = 5$ and $LightRole(LR)$, $load = 2$
- $\alpha = 0.75 \Rightarrow mwt_{a_{1.1}} = mwt_{a_{1.2}} = 5$

Well balanced growth - Example 2

Example 2 of well-balanced growth

- same three initial agents a_1, $a_{1.1}$ and $a_{1.2}$; but with $mwt_{a_1} = 8$
- same three roles :$HeavyRole(HR)$, $load = 8$; $MediumRole(MR)$, $load = 5$ and $LightRole(LR)$, $load = 2$
- $\alpha = 0.75 \Rightarrow mwt_{a_{1.1}} = mwt_{a_{1.2}} = 5$
- scenario = add the roles $\{HR, MR, MR, LR, LR, LR, LR, MR, MR, LR, LR, LR\}$ to a_1

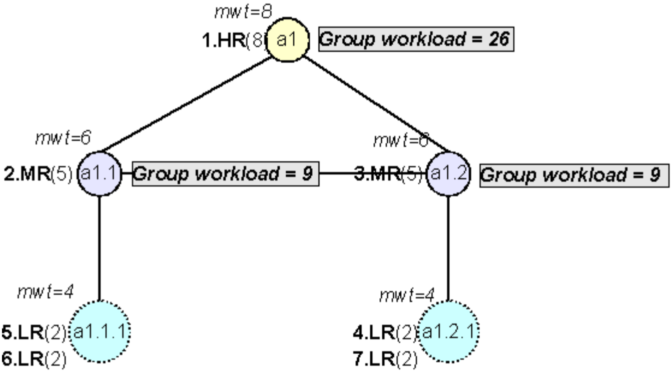
Well balanced growth - Example 2



```
a1: {HR, MR, MR, LR, LR, LR, LR, MR, MR, LR, LR, LR}.
    { 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12}.
```

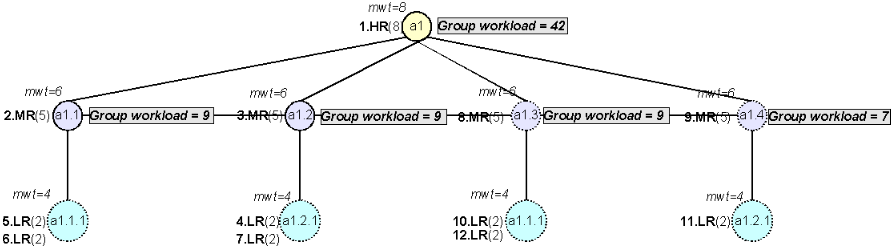
Well balanced growth - Example 2

```
a1: {HR, MR, MR, LR, LR, LR, LR, MR, MR, LR, LR, LR}.
    { 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12}.
```



Well balanced growth - Example 2

```
a1: {HR, MR, MR, LR, LR, LR, LR, MR, MR, LR, LR, LR}.
    { 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12}.
```



Implementation of an application

Implementation of a new application

Implementation of an application

Implementation of a new application

- Just have to program the behaviours

Implementation of an application

Implementation of a new application

- Just have to program the behaviours
- Relation between roles and behaviours, agents and roles and between agents are described in a xml file

Implementation of an application

Implementation of a new application

- Just have to program the behaviours
- Relation between roles and behaviours, agents and roles and between agents are described in a xml file
- An application deploys the agents on the net and transmits them roles and initial data

Implementation of an application

Implementation of a new application

- Just have to program the behaviours
- Relation between roles and behaviours, agents and roles and between agents are described in a xml file
- An application deploys the agents on the net and transmits them roles and initial data
- Currently : Turn to a OWL-S representation of roles and the HoloMAS

Applications

Applications

- Assist cooperative work into a patent department [Holomas'00] and a technological watch department [AOIS'04]

Method

1. Analysis and modelling of the human organization

Applications

Applications

- Assist cooperative work into a patent department [Holomas'00] and a technological watch department [AOIS'04]

Method

1. Analysis and modelling of the human organization
2. Improvement of the human organization based on the models

Applications

Applications

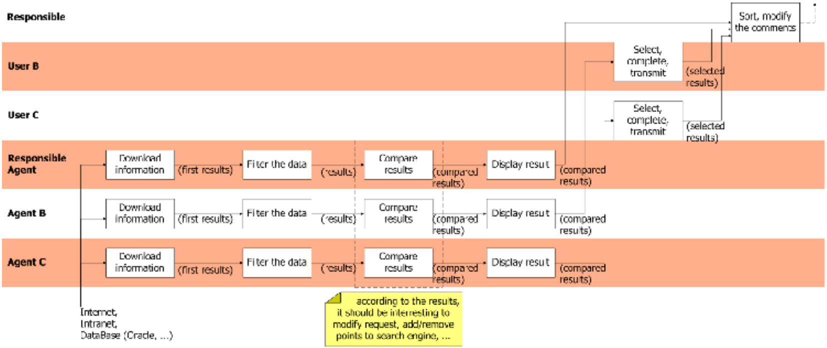
- Assist cooperative work into a patent department [Holomas'00] and a technological watch department [AOIS'04]

Method

1. Analysis and modelling of the human organization
2. Improvement of the human organization based on the models
3. Design of a MAS based on holonic principles and modeled on the human organization

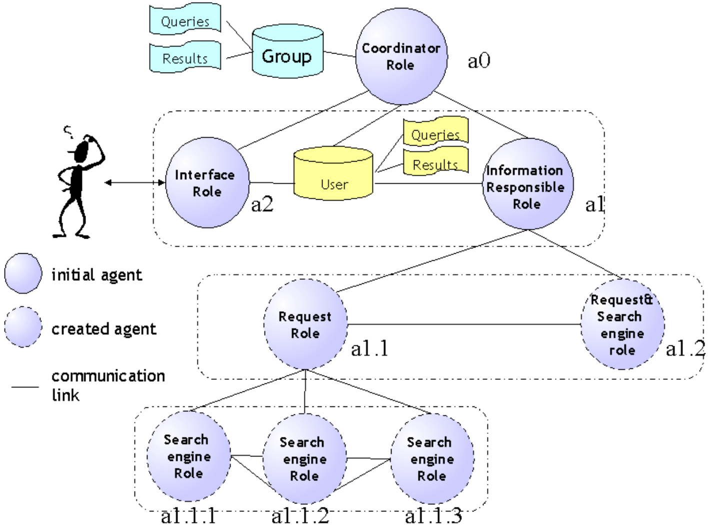
Example on a case study : Information MultiAgent System

Modeling the human activities



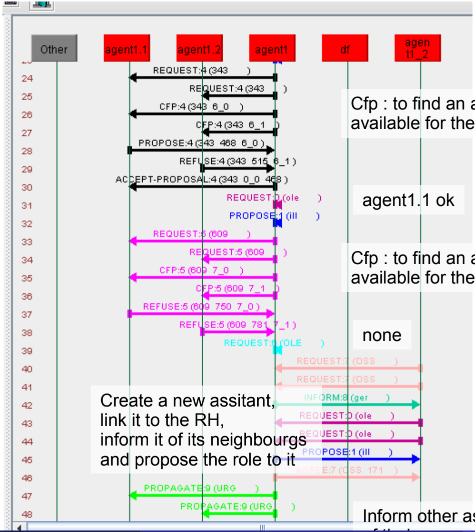
Example on a case study : Information MultiAgent System

Architecture of the proposed Holonic IMAS



Example on a case study : Information MultiAgent System

Exchanged messages during a delegation



Cfp : to find an assistant
available for the load of the role

agent1.1 ok

Cfp : to find an assistant
available for the load of the role

none

Create a new assitant,
link it to the RH,
inform it of its neighbourgs
and propose the role to it

Inform other assistants

Perspectives : self-* capacities

Perspectives

Perspectives : self-* capacities

Perspectives

- Evaluation of rules utility based on the reinforcement

Perspectives : self-* capacities

Perspectives

- Evaluation of rules utility based on the reinforcement
  - Problem : always use the same rules → to allow to test other rules

Perspectives : self-* capacities

Perspectives

- Evaluation of rules utility based on the reinforcement
  - Problem : always use the same rules → to allow to test other rules
- Criticity of the agents : only the 2 heads of the systems are said criticals

Perspectives : self-* capacities

Perspectives

- Evaluation of rules utility based on the reinforcement
  - Problem : always use the same rules → to allow to test other rules
- Criticity of the agents : only the 2 heads of the systems are said criticals
  - → use an automatic detection of the agents criticity

Perspectives : self-* capacities

Perspectives

- Evaluation of rules utility based on the reinforcement
    - Problem : always use the same rules → to allow to test other rules
- Criticity of the agents : only the 2 heads of the systems are said criticals
    - → use an automatic detection of the agents criticity
- Roles :

Perspectives : self-* capacities

Perspectives

- Evaluation of rules utility based on the reinforcement
  - Problem : always use the same rules → to allow to test other rules
- Criticity of the agents : only the 2 heads of the systems are said criticals
  - → use an automatic detection of the agents criticity
- Roles :
  - Role complexity : specified and implemented by the programmers → automatic detection of role complexity

Perspectives : self-* capacities

Perspectives

- Evaluation of rules utility based on the reinforcement
  - Problem : always use the same rules → to allow to test other rules
- Criticity of the agents : only the 2 heads of the systems are said criticals
  - → use an automatic detection of the agents criticity
- Roles :
  - Role complexity : specified and implemented by the programmers → automatic detection of role complexity
  - Perspective : automatic detection of the inter-blocking situations / non cooperative behaviour

Perspectives : self-* capacities

Perspectives

- Evaluation of rules utility based on the reinforcement
  - Problem : always use the same rules → to allow to test other rules
- Criticity of the agents : only the 2 heads of the systems are said criticals
  - → use an automatic detection of the agents criticity
- Roles :
  - Role complexity : specified and implemented by the programmers → automatic detection of role complexity
  - Perspective : automatic detection of the inter-blocking situations / non cooperative behaviour
  - Implementation of the totality of our proposition

Conclusion

Conclusion

Conclusion

Conclusion

- We propose a multi-agent organisation using both notion of roles and notion of hierarchy.

Conclusion

Conclusion

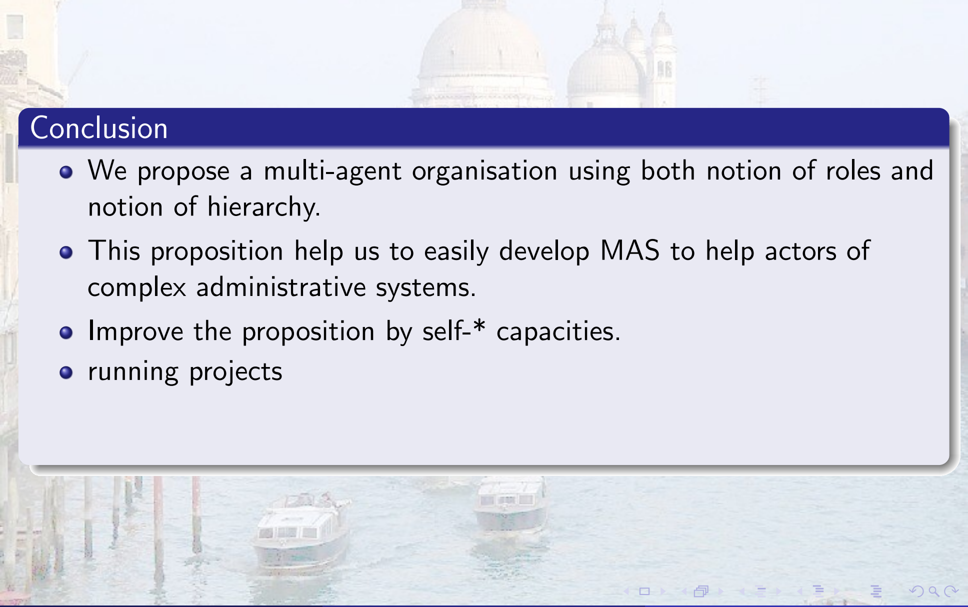
- We propose a multi-agent organisation using both notion of roles and notion of hierarchy.
- This proposition help us to easily develop MAS to help actors of complex administrative systems.

Conclusion

Conclusion

- We propose a multi-agent organisation using both notion of roles and notion of hierarchy.
- This proposition help us to easily develop MAS to help actors of complex administrative systems.
- Improve the proposition by self-* capacities.

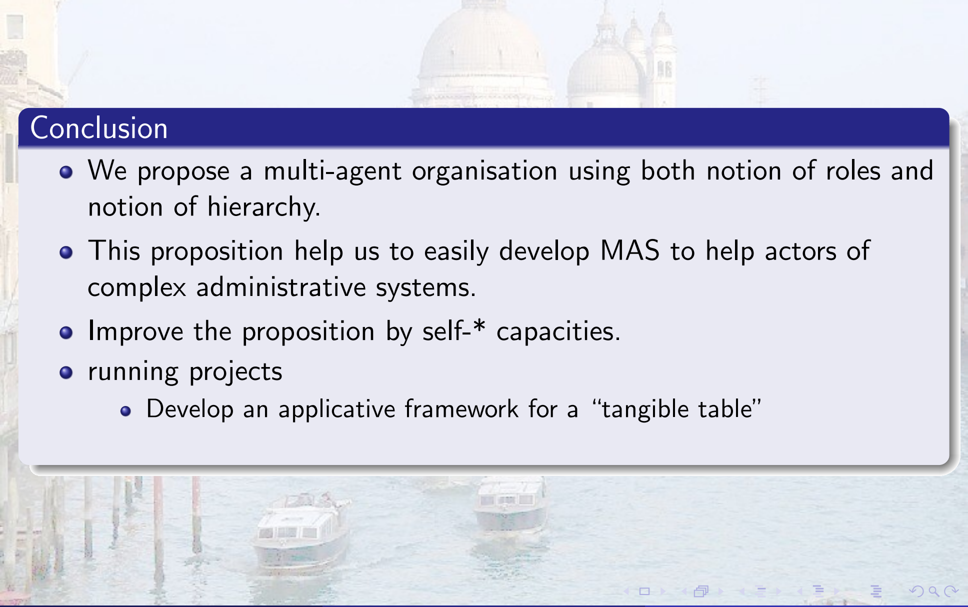
Conclusion



Conclusion

- We propose a multi-agent organisation using both notion of roles and notion of hierarchy.
- This proposition help us to easily develop MAS to help actors of complex administrative systems.
- Improve the proposition by self-* capacities.
- running projects

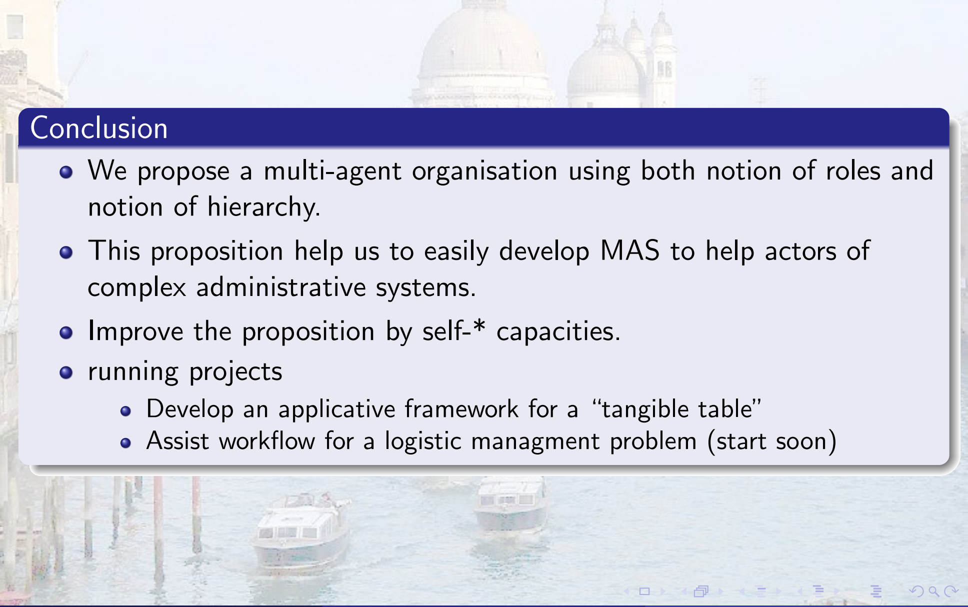
Conclusion



Conclusion

- We propose a multi-agent organisation using both notion of roles and notion of hierarchy.
- This proposition help us to easily develop MAS to help actors of complex administrative systems.
- Improve the proposition by self-* capacities.
- running projects
    - Develop an applicative framework for a "tangible table"

Conclusion



Conclusion

- We propose a multi-agent organisation using both notion of roles and notion of hierarchy.
- This proposition help us to easily develop MAS to help actors of complex administrative systems.
- Improve the proposition by self-* capacities.
- running projects
  - Develop an applicative framework for a “tangible table”
  - Assist workflow for a logistic managment problem (start soon)