An analysis and design concept for self-organization in Holonic Multi-Agent Systems

Sebastian Rodriguez, Nicolas Gaud, Vincent Hilaire, Stéphane Galland, and Abderrafiâa Koukam

Multiagent Systems and Applications Group,
Systems and Transportation Laboratory (SeT),
University of Technology of Belfort-Montbéliard (UTBM),
F-90 000 Belfort, France.
sebastian.rodriguez@utbm.fr
`http://set.utbm.fr/info/`

Abstract. Holonic Multi-Agent Systems (HMAS) are a convenient way to engineer complex and open systems. HMAS are based upon self-similar entities, called holons, which define an organizational structure called holarchy. An open issue of HMAS is to give holons means of self-organization to satisfy their goals. Our works focus on modeling and engineering of complex systems using a holonic organizational approach. This paper introduces the concept of *capacity* as the description of agents know-how. This concept allows the representation and reasoning about agents know-hows. Even more, it encourages a reusable modeling and provides agents with means to self-organize.

1 Introduction

Software agents and multi-agents systems (MAS in the sequel) are recognized as both abstractions and effective technologies for modelling and building complex distributed applications. However, they are still difficult to engineer. The current practice of MAS design tends to be limited to individual agents and small face-to-face groups of agents that operate in closed systems [13]. However, MAS aim large scale systems operating in open environments. Moreover, agents are expected to organize and cooperate in order to fullfill system's goals. It seems improbable that a rigid unscalabe organization could handle real world problems in this context. The holonic paradigm [7] has proven to be an effective solution to several problems with such complex underlying organizations [10,21,22]. Holons are defined as self-similar structure composed of holons as substructure. They are neither parts nor wholes in an absolute sense. The organizational structure defined by holons, called holarchy, allows the modelling at several granularity levels. Each level corresponds to a group of interacting holons. One issue is that holons need a representation of their know-hows in order to efficiently group, cooperate and achieve their respective goals.

Published in the 4th International Workshop on Engineering Self-Organizing Applications (ESOA'06), pp.62-75. Eds. Sven Brueckner, Salima Hassas, Mark Jelasity, Danial Yamins. May 2006. Japan.
2006 © Springer-Verlag
Republished in Lecture Notes in Artificial Intelligence (LNAI), vol. 4335, pp.15-27.
2007 © Springer-Verlag

In this paper, we introduce the notion of capacity as a description of a know-how or a service. We define this notion and integrate it into a holonic framework to enable holons to find the right holon to cooperate with.

This paper is organized as follows : secion 2 introduces the holonic framework we use and defines the concept of capacity. Section 3 shows how to use capacities and finally section 4 concludes and future research directions are presented.

2 Concepts

2.1 An organizational approach for holonic systems

A holon is a self-similar structure composed of holons as sub-structures. This hierarchical structure composed of holons is called a *holarchy*. A holon can be seen, depending on the level of observation, either as an autonomous "atomic" entity or as an organization of holons. This duality is sometimes called the *Janus Effect*[1], in reference to the two *faces of a holon*. A holon is a whole-part construct that is composed of other holons, but it is, at the same time, a component of a higher level holon. Examples of holarchies can be found in every-day life. Probably the most widely used example is the human body. The body cannot be considered as a whole in an absolute sense. It is, in fact, composed of organs, that in turn are composed of cells, molecules, etc.

Holonic Systems have been applied to a wide range of applications. Thus it is not surprising that a number of models and framework have been proposed for these systems[11,22,23]. However, most of them are strongly attached to their domain of application and use specific agent architectures. In order to allow modular and reusable modelling that minimizes the impact on the underlying architecture we propose a framework based on an organizational approach. We have selected the Role-Interaction-Organization (RIO) model [5] to represent organizations. We have leaned for this model since it enables formal specification, animations and proofs based on the OZS formalism [4].

In order to maintain this framework generic, we need to distinguish between two aspects that overlap in a holon. The first is directly related to the holonic character of the entity, i.e. a holon (super-holon) is composed of other holons (sub-holons or members). This aspect is common to every holons, thus called *holonic* aspect. And the second is related to the problem the members are trying to solve, and thus specific to the application or domain of application.

A super-holon is an entity in its own right, but it is composed by its members. Then, we need to consider how members organize and manage the super-holon. This constitutes the first aspect of the holonic framework. To describe this aspect, we define a particular organization called *Holonic Organization*. We have adopted the *moderated group*[3] as management structure of the super-holon, due to the wide range of configurations it allows. In a moderated group, a subset

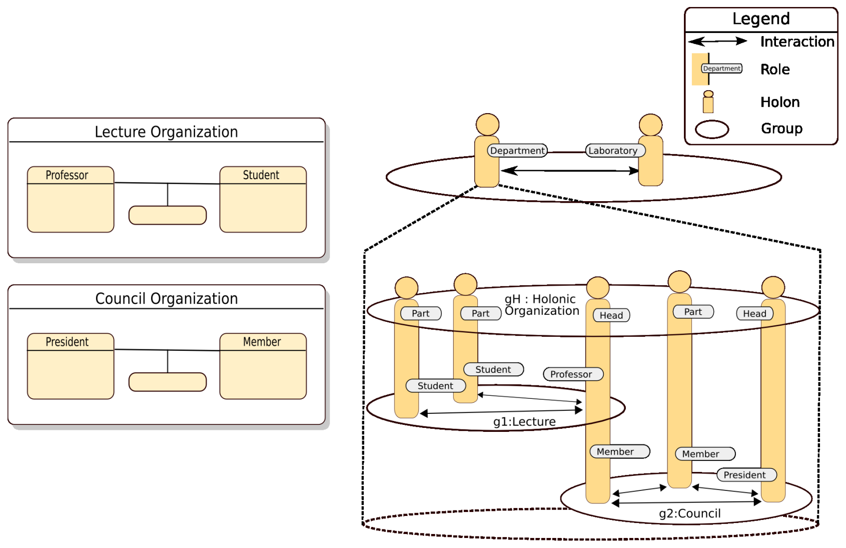
[1] Roman god with two faces. Janus was the god of gates and doorways, custodian of the universe and god of beginnings

of the members, namely *heads*, will represent all the sub-holons with the outside world.

The *Holonic Organization* represents a *moderated group* in terms of roles and their interactions. To describe the status of a member inside a super-holon, it defines three main roles: The *Head* role players are the *representatives* or *moderators* of the group, and a part of the visible interface. For the represented members we define two different roles. The *Part* role represents members belonging to only one super-holon. The *Multi-Part* role is played by sub-holons shared by more than one super-holon.

In our approach, every super-holon must contain at least one instance of the *Holonic Organization*. Every sub-holon must play at least one role of this organization to define its status in the composition of the super-holon.

Super-holons are created with an objective and to perform certain tasks. To achieve these goals/tasks, the members must interact and coordinate their actions. Our framework also offers means to model this second aspect of the super-holons. This goal-dependent interactions are modeled using organizations. We give them the name of *Internal Organizations*, since they are specific to each holon and its goals/tasks. The behaviors and interactions of the members can thus be described independently of their roles as a component of the super-holon. The set of *internal organizations* can be dynamically updated to describe additional behaviors. The only strictly required organization is the Holonic organization that describes member's status in the super-holon.



(a) Internal Organizations (b) Holonic and Internal Groups

Fig. 1: Computer Science Department Holon

This approach guarantees a clear separation between the management of the super-holon and the goal-specific behaviors and favors modularity and reusability.

For example, lets consider a department in a university. The holonic aspect makes reference to the fact that the students and teachers compose and manage the department. This is modeled using the *Holonic Organization*. On the other hand, the department is created with a specific purpose and, thus, to fulfill precise goals/tasks in the system. How members coordinate and interact to achieve these goals is modeled using the *internal organizations*. In the department example, we use two organizations: *Lecture* and *Council*. The RIO diagrams of these organizations are shown in figure 1(a).

At the holon level, an organization is instanciated into *groups*. In our example, the Department Holon is decomposed into three groups. The first represents an instance of the *Holonic Organization*. The other two groups instanciate the goals-dependent organizations : *Lecture* and *Council*. The notation *g1:Lecture* denotes that the *g1* group is an instance of the organization *Lecture*. A holon may contain several instances of the same organization.

Further details on the framework can be found in [14,16]. A formal specification of the roles described above can be found in [15].

2.2 Capacity

Large scale systems are expected to organize and cooperate in open environments. To satisfy their needs and goals, agents often have to collaborate. Thus an agent has to be able to estimate the competences of its future partners to identify the most appropriate collaborator. We have introduced the notion of capacity to deal with this issue. The capacity allows to represent the competences of an agent or a set of agents.

Definition A capacity is a description of a know-how/service. This description contains at least a name identifying the capacity and the set of its *input* and *output* variables which may have default values. The *requires* field defines the constraints that should be verified to guarantee the expected behavior of the capacity. Then the *ensures* field describe what properties the capacity guarantees if *requires* is satisfied. Finally we add a textual description to informally describe the behavior of the capacity.

In our model the capacity can thus be represented using the structure presented in figure 2 (inspired by [19] and [12]) :

By logical constraints we refer to pre/post conditions on operations and invariants on states. For example, lets consider a capacity called *FindShortestPath*. This capacity finds the shortest path in a weighted directed graph $\mathcal{G}$ from a source node s to a destination d. The description of this capacity can be stated as depicted in the figure 3. This capacity takes as input : a directed graph $\mathcal{G}$ consisting of nodes N and edges E valued by a weight function w, a source and a destination node. The output produced by this capacity, P, consists in a

Name : the name of the capacity
Input : the declaration of input variables, their type and possibly a default value.
Output : the declaration of output variables, their type and possibly an expected value for input default value.
Requires : Logical constraints defined on input variables
Ensures : Logical constraints defined on output variables
Textual Description : A textual description of the capacity

Fig. 2: The general structure of a capacity

sequence of nodes. The *requires* clause states that the node and edge sets must not be empty. It also impose that the source and destination nodes belong to the graph nodes and that the weight function gives only positive values. The *ensures* clause says that there cannot be a shorter path than P from s to d.

Name : FindShortestPath
Input :

- $\mathcal{G} = (N, E)$, directed graph. $E = N \times N$
- $w : E \rightarrow \mathbb{R}$, weight function.
- $s \in N$, source node.
- $d \in N$, destination node.

Output : $P = \langle s = i_0, i_1, \cdots, i_{n-1}, d = i_n \rangle$, with $\forall\, k \in \{0..n\}, i_k \in N$ the shortest path P between s and d.
Requires : $N \neq \varnothing$ and $E \neq \varnothing$ and $\forall (u, v) \in E / w(u, v) \geq 0$
Ensures : $\forall\, j_t \in N,\ t \in \{0..m\}$

$$\nexists\ Q = \langle s = j_0, j_1 \cdots, j_m = d \rangle /$$
$$P \neq Q \wedge \sum_{t=0}^{m-1} w(j_t, j_{t+1}) < \sum_{k=0}^{n-1} w(i_k, i_{k+1})$$

There exists no path Q in the graph linking s to d shorter than P.
Textual Description : provides a solution to the single-source shortest path problem for a directed graph with non-negative edge weights.

Fig. 3: The *FindShortestPath* capacity

The definition of the capacity doesn't include any references to entities exhibiting this know-how/service. Indeed, we want to clearly separate the capacity of how it is realized.

However, from the super-holon point of view, we can categorize its capacities in three subcategories:

Atomic The capacity is already present in one of the members of the super-holon. In this case, the head has to simply request the member possessing the required capacity to perform it.
Liaised The capacity is obtained from a subset of the member's capacities following a known protocol.
Emergent The capacity is not present as an atomic capacity nor it can be obtained as composition of them. The capacity *emerges* from the interactions of the members.

The capacity is *atomic* for the super-holon if one of the members provides the capacity, but it does not have any implications on how this member obtains this capacity. This taxonomy of capacity is only relative to the super-holon point of view. The distinction between the capacity and the means to obtain it, and how we have integrated this concept into our holonic organizational model, will be detailed in the next section.

2.3 Integrating capacities into a holonic organizational perspective

As we already mentioned, we use an organizational approach to model holonic MAS. We propose an extension of the RIO model[5] to integrate the concepts of *Holon* and *Capacity*. An overview of this meta-model is presented in figure 4 using an UML-like diagram.

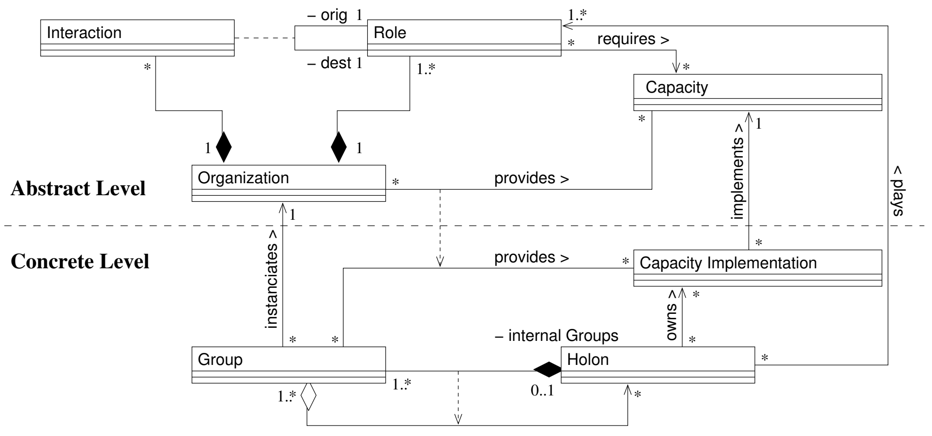


Fig. 4: Overview of the Organizational Meta-Model

As in RIO, the behaviors of the members are specified in terms of roles and their interactions. A role is defined as the abstraction of a behavior or/and a status in an organization. An interaction is a link between two roles such that an action in the first role produces a reaction on the second one. An organization is defined by a set of roles, their interactions and a common context. Finally a capacity is defined as presented in the previous section.

To obtain a generic model of organization, we need then to define a role without making any assumptions on the architecture of the holon which will play this role. Basing the description of these behaviors (Roles) on capacities enables a modular and reusable modeling of holonic MAS. Indeed, capacities describe what the holon is capable of doing (Abstract Level), independently of how it does it (Concrete Level). So to catch these two different levels of abstraction in our organizational model, we introduce the notion of capacity implementation. The capacity is the description of a given competence/service, while its implementations are the way to obtain that competence or service. For the example of the capacity *FindShortestPath*, the *Dijkstra*'s and the *Bellman-Ford* algorithms constitute two available implementations. From a programming point of view, the notion of capacity implementation promotes re-usability and modularity and in this sense can be considered as a basic software component. Multiple implementations can be associated to a single capacity.

A role defines a behavior based on what the holon is capable of doing (i.e. the holon's capacities). Thus, a role requires that the role player has specific capacities. A holon has to possess all capacities required by a role to play that role.

On the other hand, a role confers to its player a certain status in the organization and the right to perform its capacities. A role thus confers holon the authorization to wield some of its capacities in the context defined by the organization. This context is materialized at the Concrete Level by the Group. A group represents then an instance of an organization. A holon belonging to a group must play at least one role in this group. A holon can belong to several groups.

In addition a holon may be composed of groups. A super-holon contains at least the holonic group and possibly a set of internal groups, instances of internal organizations. Of course a super-holon cannot be a member of one of its internal groups.

We suppose that every holon has a set of basic capacities. It may also have a set of specific capacities (e.g. *FindShortestPath*) that, as we will see in section 3.2, can dynamically evolve.

3 Using Capacities to enable HMAS Self-organization

3.1 Organization capacities

As described by John H Holland : "The behavior of a whole complex adaptive system[*cas*] is more than a simple sum of the behaviors of its parts; *cas* abound non linearity" [6].

The notion of capacity provides means to control and exploit these additionnal behaviors, emerging from members interactions, by considering an organization as a capacity implementation. Organizations used to model members interactions offer a simple way to represent how these capacities are obtained from the members. This becomes specially useful to represent *Liaised* and *Emergent* Capacities.

For example, we have already mentioned that the *Dijkstra*'s and *Bellman-Ford* algorithms can be two possible implementations of the capacity *FindShortestPath*. If we consider that organizations can also be seen as possible implementations, the *Ant Colony* organization may then be also considered as an implementation of the capacity *FindShortestPath*. The *Ant Colony* is a well known organization able to determine a solution to the shortest path problem in a directed graph. The solution (the shortest path) results from Ants interaction in their environment. According to the description stated at the figure 3, the environment is represented by the graph $\mathcal{G}$, the source node s is assimilited to the Ant-hill, and the destination node d to a source of food.

The figure 5 shows an example of three holons in interactions. Holon 1, playing the *Route Requester* role, is looking for the shortest route to a given destination, and thus asks the *Route Providers*. The behavior, described by the *Route Provider* role, is based on the assumption that the role player has the *FindShortestPath* capacity. As long as the implementation honors the constraints established by the capacity, the holon is authorized to play the role. In our example, two implementations are present. The holon 2 owns an implementation based on the *Dijkstra*'s algorithm while holon 3 obtains its capacity through an *Ant Colony*. Holon 3 contains an instance of the *Ant Colony* organization (depicted in figure 6) noted $g1 : Ant\ Colony$. This denomination indicates that group $g1$ is an instance of the *Ant Colony* organization. As a such, members involved in the group play one of the roles defined on the RIO diagram given at figure 6.

The fact that an organization can provide a capacity takes all its sense when we associate it to the holonic vision. Because the super-holon can exploit the additional behavior emerging from its members interactions and so play roles inaccessible to its members. It remains one more issue in order to integrate it into the holonic modelling : how to map the external stimuli of the super-holon to the actions and capacities of the members. The *head* represents the solution to this problem. The members playing this role are part of the interface of the super-holon. Thus, in charge of redistributing or translating the external incoming information to the other members of the holon (playing *Part* and *MultiPart* roles). Certain organizations may require to be adapted to this mode of representation. For the *Ant Colony* organization (cf. figure 6), a special role : the *Supervisor*, played by the *Head*, have been added. First of all, the *Supervisor* is in charge of initializing the environment of the colony with the specified input graph $\mathcal{G}$ and emitting the signal to launch the Ants. Then it observes the *Ant Colony* to determine when the result is available and forward it to its super-holon. This result being emergent, the presence of an observer to determine the availablity of the result is imperative. The holon 3 in figure 5 contains a group $g1$ which is an instance of the *Ant Colony* Organization.

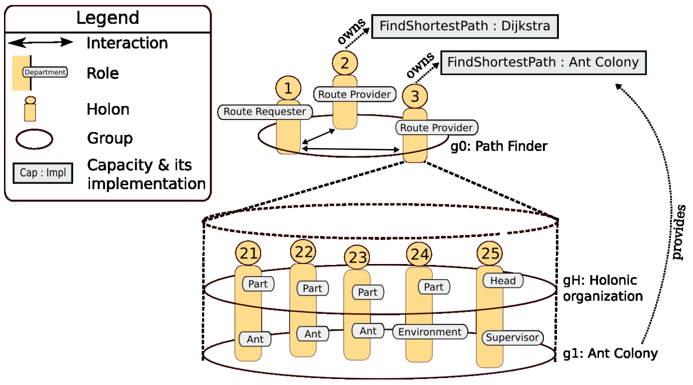


Fig. 5: Atomic or Emergent Capacity

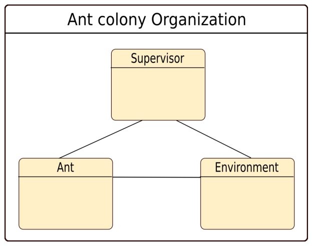


Fig. 6: RIO Diagram of the *Ant Colony* Organization

In this sense, an organisation can, under certain conditions, provide one or more capacities. Thus, it represents a way to implement or obtain a capacity. This feature can be exploited in the Analysis and Design phase. To that end, the capacities provided by an organization have to be added to its description (especially in Organizational Design Pattern).

The self-organization mechanisms enabled by the notion of capacity will be detailed in the following section.

3.2 Capacity Dynamics

As we have already precised, each holon originally possess a set of capacities that can dynamically evolve. In order to acquire a new capacity a holon may instanciate a new internal organization providing the required capacity. This process can be summarized by the following steps and is depicted in figure 7 :

1. First, the holon tries to match the capacities provided in organizations' descriptions with the required capacity. To assure this matchmaking process in an open system a common description language is required to match holon capacites, [19,20] propose a model to deal with dynamic service matchmaking that can be easily adapted to our case.
2. If matches are found the holon has to choose among the different organizations the one which seems the fittest. This choice is essentially based on the capacities required by the chosen organization and the already present member's capacities.
3. When an organization has been chosen the holon has yet to instantiate the defined roles and interactions. Either the chosen organization's roles are played by sub-holons member, or it has to recruit new members capable of playing those roles.
4. When each role defined in the chosen organization has a player in the freshly instanciated group. The super-holon is able to obtain the capacity implementation and can thus play new roles.

To illustrate these steps, lets now consider our *FindShortestPath* capacity example. We suppose that the group g_0 of figure 5 contains a *Route Requester* and only one *Route Provider*. The holon 3 wish integrate this group as *Route Provider* but it doesn't possess the required capacity *FindShortestPath* (cf. fig. 7, step 1).

It has thus two possibilities. First, the recruitment of a member, already owning the *FindShortestPath* capacity. In this case it would also obtain it as an atomic capacity.

Second, instanciate an organization able to provide it, like the *Ant Colony* organization. This organization is found using a matchmaking process (cf. fig. 7, step 2). Lets condiser the situation where holon 3 choses this alternative and integrates an additional internal group, instanciating the *Ant Colony* organization (cf. fig. 7, step 3). It recruits new members able to play the various role of this organization (cf. fig 6).

Owning henceforth the required capacity, it's able to play the *Route Provider* role and thus joins the group g_0 (cf. fig. 7, step 4).

4 Related Works

Several approaches related to agent capabilities have been already proposed in various domains of MAS.

In the domain of Semantic Web and Web Agents, [19,20] propose an Agent Capability Description Language (LARKS) and discuss the Service Matchmaking process using it. Thus a first description of Agent Capability using LARKS is given. However this decription is only used in the Service Matchmaking process and not used during the analysis nor the modelling phases. These aspects are

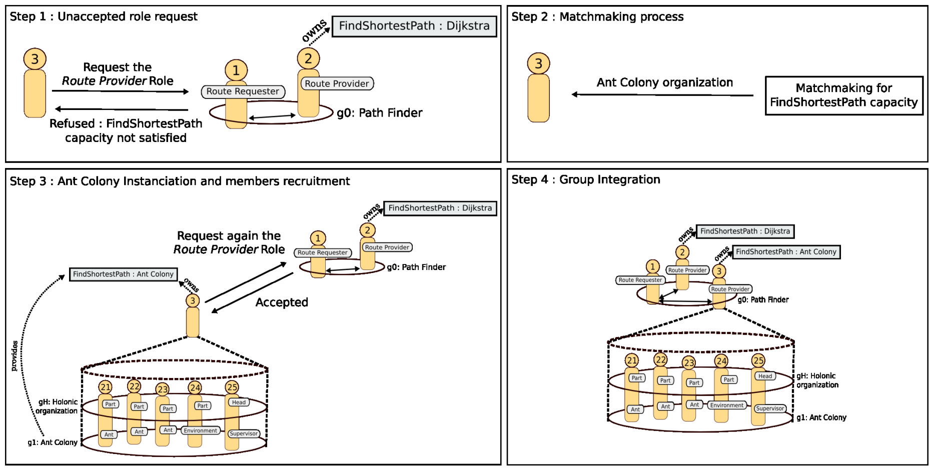

Step 1 : Unaccepted role request
3
Request the *Route Provider* Role
Refused : FindShortestPath capacity not satisfied
1
Route Requester
2
Route Provider
owns
FindShortestPath : Dijkstra
g0: Path Finder
Step 2 : Matchmaking process
3
Ant Colony organization
Matchmaking for FindShortestPath capacity
Step 3 : Ant Colony Instanciation and members recruitment
Request again the *Route Provider* Role
Accepted
1
Route Requester
2
Route Provider
owns
FindShortestPath : Dijkstra
g0: Path Finder
FindShortestPath : Ant Colony
owns
3
provides
21
22
23
24
25
Part
Part
Part
Part
Head
Ant
Ant
Ant
Environment
Supervisor
gH: Holonic organization
g1: Ant Colony
Step 4 : Group Integration
owns
FindShortestPath : Dijkstra
owns
FindShortestPath : Ant Colony
1
2
3
Route Requester
Route Provider
Route Provider
g0: Path Finder
21
22
23
24
25
Part
Part
Part
Part
Head
Ant
Ant
Ant
Environment
Supervisor
gH: Holonic organization
g1: Ant Colony

Fig. 7: Acquiring a new capacity by integrating a new internal organization

tackled in our approach with the notion of capacity as a basic decription of an agent know-how.

To distinguish the agent from its competences, [17] and [18] have introduced the notion of *skill* to describe basic agent abilities and allowing the definition of an atomic agent, that can dynamically evolve by learning/acquiring new *skills*. Then [1,2] have extended this approach to integrate this notion of *skill* as a basic building block for role specification. [8,9] also consider agent capability as a basic building block for role specification in their meta-model for MAS modeling. But these capabilities are inherent to particular agents, and thus to specific architectures. In these models the role is considered as a link between agents and a collection of behaviors embodied by the skills. This really differs from our view of the notion of role. For us a role is a first class entity, the abstraction of a behavior or/and a status in an organization (extension of [5]), that should be specified without making any assumptions on its susceptible players. In other words, in these approaches, the skill is directly related to the way to obtain a service, and thus represents a basic software component. However, the description of the general class of related services and the fact that a given agent ability can be obtained by various implementations is not developed. We can thus consider that these aspects are captured in our model with the notion of capacity implementation.

In a more general way, we can consider that our approach is situated in the confluence of these various models, linking the description of an agent capability and its various possible implementations. We thus provide agents with means to reason about their needs/goals and to identify the way to satisfy/achieve them. We thus benefit of the advantages of both approaches, increasing reuasibility and modularity by separating the agent from its capacities, and the capacity from its various implementations.

Considering an organization as a possible capacity implementation constitutes our main contribution. We can thus consider that a group of interacting agent can provide a capacity to an upper level. This takes all its interest in the case of holonic MAS, where the super-holon can exploit additionnal behaviors emerging from members interactions to obtain a new capacity. In a same way, we also provide a modeling tool to deal with intrinsic emergent properties of a system and catch them directly from the analysis phase.

5 Conclusion

In this paper, we have presented the concept of capacity to enable a modular and reusable model of organizations. To achieve that, the role specification is based on the description of required know-hows, described using capacities. To play a role, a holon has to possess an implementation (that could be specific according to its architecture) for each required capacity.

By considering, an organization as a possible capacity implementation, we provide means to exploit additional behaviors emerging from a group of agents in interaction. Combining this representation with the holonic approach, the super-holon, by instanciating specific organizations, can obtain new capacities from the collaboration of its members and therefore play roles, inaccessible to its members as individuals.

Finally we introduce self-organization mechanisms allowing a holon to dynamically change its set of capacities and so achieve its new goals.

This work is part of larger effort to define a well founded framework for Holonic MAS applications. Future research will deepen the formal specification of capacities and define a matchmaking process to find capacities implementations.

References

1. Emmanuel Adam and René Mandiau. A hierarchical and by role multi-agent organization: Application to the information retrieval. In *ISSADS*, pages 291–300, 2005.
2. Emmanuel Adam and René Mandiau. Roles and hierarchy in multi-agent organizations. In M. Pechoucek, P. Petta, and L.z. Varga, editors, *4th International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2005*, number 3690 in LNAI, pages 539–542, Budapest, Hungary, September 2005. Springer-Verlag.
3. Christian Gerber, Jörg H. Siekmann, and Gero Vierke. Holonic multi-agent systems. Technical Report DFKI-RR-99-03, Deutsches Forschungszentrum für Künztliche Inteligenz - GmbH, Postfach 20 80, 67608 Kaiserslautern, FRG, May 1999.
4. Pablo Gruer, Vincent Hilaire, Abder Koukam, and P. Rovarini. Heterogeneous formal specification based on object-z and statecharts: semantics and verification. *Journal of Systems and Software*, 70(1-2):95–105, 2004.
5. Vincent Hilaire, Abder Koukam, Pablo Gruer, and Jean-Pierre Müller. Formal specification and prototyping of multi-agent systems. In Andrea Omicini, Robert

Tolksdorf, and Franco Zambonelli, editors, *Engineering Societies in the Agents' World*, number 1972 in Lecture Notes in Artificial Intelligence. Springer Verlag, 2000.

6. John H. Holland. *Hidden order: how adaptation builds complexity*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 1995.
7. Arthur Koestler. *The Ghost in the Machine*. Hutchinson, 1967.
8. Eric Matson and Scott A. DeLoach. Autonomous organization-based adaptive information systems. In *IEEE International Conference on Knowledge Intensive Multiagent Systems (KIMAS '05)*, Waltham, MA, April 2005.
9. Eric Matson and Scott A. DeLoach. Formal transition in agent organizations. In *IEEE International Conference on Knowledge Intensive Multiagent Systems (KIMAS '05)*, Waltham, MA, April 2005.
10. F. Maturana, W. Shen, and D. Norrie. Metamorph: An adaptive agent-based architecture for intelligent manufacturing, 1999.
11. Francisco Maturana, Weiming Shen, and Douglas Norrie. Metamorph: An adaptive agent-based architecture for intelligent manufacturing. *International Journal of Production Research*, 37(10), 1999. 2159-2174.
12. B. Meyer. Applying 'design by contract'. *IEEE Computer*, 25(10):40–51, October 1992.
13. James Odell, Marian Nodine, and Renato Levy. A metamodel for agents, roles, and groups. In James Odell, P. Giorgini, and Jg Mller, editors, *Agent-Oriented Software Engineering (AOSE) IV*, Lecture Notes on Computer Science. Springer, 2005.
14. Sebastian Rodriguez, Vincent Hilaire, and Abder Koukam. Towards a methodological framework for holonic multi-agent systems. In *Fourth International Workshop of Engineering Societies in the Agents World*, pages 29–31, Imperial College London, UK (EU), October 2003.
15. Sebastian Rodriguez, Vincent Hilaire, and Abder Koukam. Fomal specification of holonic multi-agent system framework. In *Intelligent Agents in Computing Systems, International Conference on Computational Science (3)*, number 3516 in LNCS, pages 719–726, Atlanta, USA, 2005.
16. Sebastian A. Rodriguez. *From analysis to design of Holonic Multi-Agent Systems: a Framework, methodological guidelines and applications*. PhD thesis, Université de Technologie de Belfort-Montbéliard, 2005.
17. JC. Routier, P. Mathieu, and Y. Secq. Dynamic skill learning: A support to agent evolution. In *AISB'01 Symposium on Adaptive Agents and Multi-Agent Systems*, pages 25–32, 2001.
18. M. Savall, M. Itmi, and J-P. Pecuchet. Yamam : a new organization model for multi-agent systems and its platform named phoenix. In *Conference SCSC 2000*, Orlando, USA, 2001.
19. K. Sycara, M. Klusch, S. Widoff, and J. Lu. Dynamic service matchmaking among agents in open information environments. *SIGMOD Record (ACM Special Interests Group on Management of Data)*, 28(1):47–53, March 1999.
20. K. Sycara, J. Lu, M. Klusch, and S. Widoff. Matchmaking among heterogeneous agents on the internet. In *Proceedings of the 1999 AAAI Spring Symposium on Intelligent Agents in Cyberspace*, March 1999.
21. F. Tecchia, C. Loscos, R. Conroy, and Y. Chrysanthou. Agent behaviour simulator (abs): A platform for urban behaviour development. In *GTEC'2001*, 2001.
22. M. Ulieru and A. Geras. Emergent holarchies for e-health applications: a case in glaucoma diagnosis. In *IECON 02 [Industrial Electronics Society, IEEE 2002 28th Annual Conference of the]*, volume 4, pages 2957– 2961, 2002.

23. J. Wyns. *Reference architecture for Holonic Manufacturing Systems - the key to support evolution and reconfiguration.* PhD thesis, Katholieke Universiteit Leuven, 1999.