# An organizational approach to engineer emergence within holarchies

Massimo Cossentino* and Stéphane Galland+ and Nicolas Gaud+ and Vincent Hilaire+ and Abderrafiaa Koukam+

December 19, 2012

* ICAR-CNR,
University of Palermo,
Viale delle Scienze, Ed. 11.
c/o CUC (Centro Universitario di Calcolo)
90128 Palermo ITALY
+ SeT laboratory,
UTBM,
90010 Belfort cedex
FRANCE

**Abstract**

An open issue in self-organization is how to engineer emergent behaviors. This issue is also of interest for engineering holonic multi-agent systems as any level of a holarchy is dependant of the emergent behaviors of its sub-levels. In order to tackle this specific feature of holonic multi-agent systems, the capacity concept which abstracts a know-how from its concrete realization is introduced. The use of this concept is illustrated in this paper through a case study using the ASPECS development process which enables the analysis, design, implementation and deployment of holonic multi-agent systems and integrates the capacity as a core concept of it's underlying metamodel, called CRIO (i.e Capacity-Role-Interaction-Organisation).

## 1 Introduction

Works on self-organisation and emergence are common in the Multi-Agent Systems (MAS) field [33]. As pointed out in [34] a still open issue is how to engineer self-organizing applications or in other words how to define a global goal, and to design local behaviors so that a global behavior, able to satisfy this goal, emerges. In [29] the capacity concept is proposed as an abstraction that allows agents and holons to exhibit self-organisation features. In this paper the way in which such an abstraction may be useful to engineer self-adaptation and

self-organisation within a Holonic MAS (HMAS) dedicated to the simulation of robot soccer games is shown and exemplified all along the software development process.

In multi-agent systems, the vision of holons is someway closer to the one that MAS researchers have of *Recursive* or *Composed* agents. A holon constitutes a way to gather local and global, individual and collective points of view. A holon is a self-similar structure composed of holons as sub-structures and a hierarchical structure composed of holons is called a *holarchy*. A holon can be seen, depending on the level of observation, either as an autonomous whole entity or as an organisation of holons (this is often called the *Janus effect*, in reference to the two *faces of a holon* [22]).

This abstraction is integrated in a complete software development process from requirements to code, namely ASPECS [1][8]. ASPECS is based on a holonic organizational metamodel and provides a step-by-step guide from requirements to code allowing the modeling of a system at different levels of details using a suite of refinement methods. Using a holonic perspective, the designer can model a system with entities of different granularities. He can recursively model components and sub-components of a system until he achieves a stage where the requested tasks are manageable by atomic easy-to-implement entities.

This paper is organized as follow: related works are discussed in section 2, section 3 discusses the methodological background based on our holonic framework and the ASPECS development process. Section 4 describes the various faces of the concept of capacity all along this process. Section 5 presents the robot soccer simulation model used to illustrate the emergence control in a holarchy. Finally some conclusions are drawn in section 6.

## 2 Related Works

Emergence phenomenon in artificial systems has already been studied as in [14]. This reference gives a definition:

- a collection of interactive agents: the process;
- an epiphenomenon produced by this process at the macro level;
- a natural interpretation of this epiphenomenon as computation or computation results.

Other works distinguish between several levels of emergence from weak to strong [9]. The common point of these approaches is the existence of, at least, two levels. This feature is particularly interesting for the design of holarchies where each level relies on the behaviour of it lower levels.
Several approaches related to the exploitation of agent capabilities have been already proposed in various domains of MAS.

[1] ASPECS: Agent-oriented Software Process for Engineering Complex Systems, see `http://www.aspecs.org/`

In [26] the authors propose definitions for swarming and self-organized MAS and give general principles on how to engineer swarming MAS. These principles are illustrated through several case studies. Even if it is a very important work both on theoretical and practical aspects, the given principles are not well formalized and can be ambiguous. Their application may lead to several different results. Moreover, the approach presented in this paper is based on organizational concepts which enable the definition of ODP (Organisational Design Pattern) from swarming metaphors. These ODP can be easily reused as many MAS methodologies and development platforms integrate organizational concepts.
The authors of [36] deal with the engineering of emergence as it appears in many biological systems. The underlying assumption is that there must exist several levels and/or timescales to explain and describe emergence. Their proposition consists in a three elements architecture to be refined for any specific problem. These elements are: the System of System (SoS) model that describes the high level system, the local model that describes the lower level and the the integrated model that describes an integration environment between these two levels. However, the work presented in [36] only sketch the architecture and do not propose a methodology to apply it.

In the domain of Semantic Web and Web Agents, [37, 38] 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 description is only used in the Service Matchmaking process and not used during the analysis nor the modeling phases. These aspects are tackled in our approach with the notion of capacity as a basic description of an agent know-how.

[28] discusses the concept of capability and the concept of opportunity, respectively representing the necessary and the sufficient conditions to achieve a goal. These concepts are similar to the concepts of capacity and capacity realisation but they do not take into account organisational nor holonic concerns.

To distinguish the agent from its competencies, [30] and [31] have introduced the notion of *skill* to describe basic agent abilities thus 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. [23, 24] also consider agent capability as a basic building block for role specification in their meta-model for MAS modeling. These capabilities are however 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 behaviours embodied by the skills. 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. This really differs from our view of the notion of role. For us a role is a first class entity, the abstraction of a behaviour or/and a status in an organisation (extension of [18]), that should be specified without making any assumptions on its susceptible players. It is

assumed that these aspects are captured in our model with the notion of capacity implementation.

In a more general way, our approach is situated in the confluence of these various models, thus linking the description of an agent capability with its various possible implementations. As a consequence agents are provided with means to reason about their needs/goals and to identify the way to satisfy/achieve them. The proposed approach benefits of the advantages of both approaches, thus increasing reusability and modularity by separating the agent from its capacities, as well as separating the capacity from its various implementations.

Considering an organisation as a possible capacity implementation constitutes one of our main contributions; group of interacting agents can provide a capacity to an upper level. This becomes particularly interesting in the case of holonic MAS, where the super-holon can exploit additional behaviours emerging from members interactions in order to exhibit a new capacity. In the same way, this is a modeling tool to deal with intrinsic emergent properties of a system and to catch them directly in the analysis phase.

# 3 Methodological background

In this section an overview will be provided about the concepts related to holonic systems and ASPECS. Holonic systems are particularly useful in modeling solutions to complex, hierarchical decomposable problems but they need a rigorous approach and the definition of a huge set of concepts as it will be explained in the following subsection.

## 3.1 Holonic Systems

A holon is a whole-part construct that may be composed of other holons, but it may be, at the same time, a component of higher level holons. Examples of holarchies can be found in every-day life. Probably the most widely used example is the human body. The body may be considered as a whole or as a nested hierarchy of organs composed of cells, in turn composed of molecules, etc.

Holonic Systems have been already applied to a wide range of applications and a number of models and frameworks have been proposed for these systems [25, 39, 42]. However, most of them are strongly attached to their domain of application and use specific agent architectures. In order to allow modular and reusable modeling that minimizes the impact on the underlying architecture a framework based on an organizational approach is proposed. The Capacity-Role-Interaction-Organisation (CRIO) meta-model [29] is selected to represent organizations since it enables formal specification, animations and proofs based on the OZS, which stands for Object-Z and Statecharts, formalism [17].

In order to maintain this framework generic, two aspects that overlap in a holon should be distinguished. The first is directly related to the holonic nature of the entity (i.e a holon, called super-holon, is composed of other holons, called

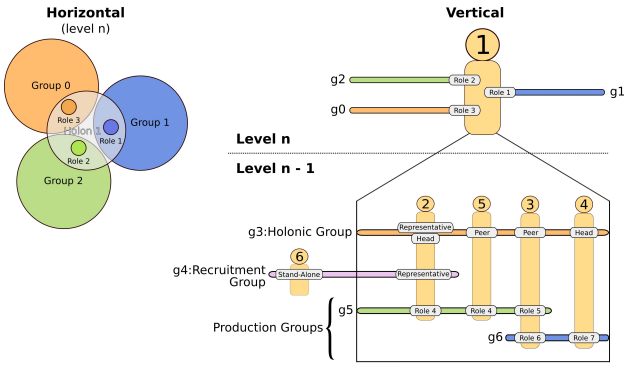


Figure 1: Horizontal and Vertical decomposition of a holon

sub-holons or members) and deals with the government and the administration of a super-holon. This aspect is common to every holon and thus called the *holonic* aspect (see figure 1). It describes the decision making process and how members organize and manage the super-holon. The second aspect is related to the problem to solve and the task to be performed. It depends on the application domain and is called the *production* aspect. It describes action coordination mechanisms and interactions between members to achieve the objectives of the super-holons.

To manage the first aspect of a composed holon, a particular organisation called *Holonic Organisation* is defined to describe the government of a holon and its structure in terms of decision making (authority, power repartition) and representative functions (interactions with other holons). In [16] three different structures are proposed. The federation where all members of the holon are equals with respect to the interactions with the outside of the holon. The fusion where members disappear to form a single entity and the *moderated group* where a subset of the members are chosen to act as mediator with the outside of the holon. In this work the moderated group is adopted as management structure of the super-holon, due to the wide range of configurations it allows. The *Holonic Organisation* represents a *moderated group* in terms of roles (called *holonic roles*) and their interactions. Three *holonic roles* have been defined to describe the status of a member inside a super-holon and one role to describe the status of non-members: (i) *Representative* is the externally visible part of a super-holon. It is an interface between the outside world (same level or upper level) and the other holon members. It may represent other members in taking decisions or accomplishing tasks (i.e., recruiting members, transfering information, etc). The *Representative* role can be played by more than one member at the same time. (ii) *Head* or decision maker represents a privileged status conferring a certain level of authority in taking decisions inside the holon. This role is not to be confused with the *Representative* role. Indeed, *Representative* role-players are not allowed to take decisions for the holon. Nevertheless, a holon can be at the same time a *Representative* and a *Head*. (iii) *Peer* is in charge of doing tasks

assigned by *Heads*. A *Peer* can also have an administrative duty, and it may be employed in the decision making process. It depends on the configuration chosen for modeling the super-holon. (iv) *Stand-Alone* or non-member represents a particular status inside a holonic system. In contrast to the previous holonic roles, it represents the way a member sees a non-member holon. *Stand-Alone* holons may interact with the *Representatives* to request their admission as new members of an existing super-holon. Admissions of new holons are problem dependant and can be delt with many different mechanisms such as vote among members, assertions on candidate capacities, etc.

The three first holonic roles describe the status of a member within a super-holon and participate in defining the holonic organisation. Each of these roles can be played by one or more members, knowing that any super-holon must have at least one *Representative* and one *Head*. The roles *Head*, *Peer* are exclusive between them, while *Representative* may be played simultaneously with one of the two others. The *Part* status represents members belonging to only one super-holon while the *Multi-Part* status represents sub-holons belonging to more than one super-holon.

In our approach, each super-holon must contain exactly one instance of the *Holonic Organisation*: the holonic group. Every sub-holon must play at least one role in this group to define its status inside the super-holon. Figure 1 illustrates these different aspects of a composed holon and describes its typical structure from a horizontal and a vertical point of view. The vertical decomposition represents the decomposition of the holon numbered 1 in the figure, in several sub-holons, here 2, 3, 4 and 5. These sub-holons constitute the level just below the level of holon 1. The horizontal decomposition represents the fact that an holon, whatever the level it belongs to, can play different roles within groups. For example, the holon 2 plays the roles *Representative* and *Head* in the *holonic group* of the holon 1 and other roles in groups 4 and 5. Holon 6 plays the role *Stand-alone* to interact with one of the *representatives* of the super-holon 1 to manage recruitment related interactions.

Super-holons are created with a goal 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, namely *Production 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. Any number of *production organizations* is possible. Each organisation describes an aspect of the problem dependant aspects. The only strictly required organisation is the *Holonic organisation* that describes member's status in the super-holon.

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

## 3.2 Overview of the ASPECS development process

The ASPECS software development process can be considered as an evolution of the PASSI [7] methodology for modeling HMAS and it also draws from experiences about holon design coming from the CRIO approach [29]. ASPECS combines an organizational approach with an holonic perspective. Its target scope can be found in complex systems and especially hierarchical complex systems. The principle of ASPECS consists in analyzing and decomposing the structure of complex systems by means of a hierarchical decomposition. The ASPECS process consists in three phases that are briefly described below. Figure 2 describes the various activities and phases of ASPECS and their associated goals. The *System Requirements Analysis* phase is based on the identification

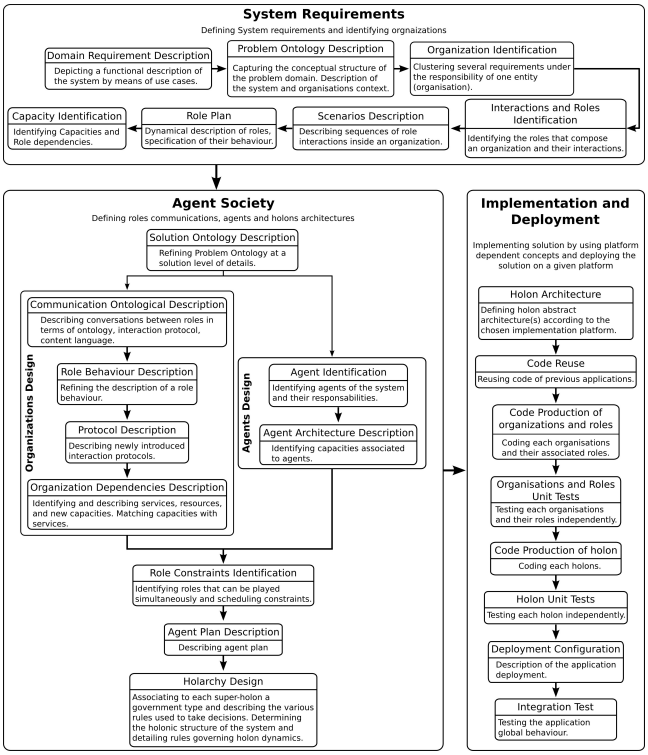


Figure 2: Overview of ASPECS process

of a hierarchy of organizations, whose global behavior may represent the system under the chosen perspective. It starts with requirements analysis that is performed by means of classical techniques such as use cases. Domain knowledge and vocabulary associated to the target application are then collected and explicitly described in the problem ontology. Each requirement is then associated to an organisation, that represents a global behavior able to fulfill the associated requirements. The context of each organisation is defined by a set of

concepts of the problem ontology. Organisation identification generates a first hierarchy of organizations; these will later be extended and updated in an iterative process in order to obtain the global organisation hierarchy representing the system structure and behaviors. Identified organizations are then decomposed into a set of interacting sub-behaviors modeled by roles. The goal of a role is to contribute to the fulfillment of (a part of) the requirements of the organisation within which it is defined. In order to design modular and reusable organisation models, roles should be specified without making any assumptions on the architecture of the agent that may play them. To meet this objective, the concept of capacity was introduced. The analysis phase ends with the capacity identification activity that aims at determining if a role requires a capacity and then adapting its behavior description. At this step a new iteration may possibly start. The analysis process may stop when all capacities required by roles at the lowest level of the hierarchy are considered to be manageable by atomic easy-to-implement entities.

The *Agent Society Design* phase aims at designing a society of agents whose global behavior is able to provide an effective solution to the problem described in the previous phases and to satisfy associated requirements. The objective is, now, to provide a model of the agent society involved in the solution in terms of social interactions and dependencies among entities (holons and/or agents). Previously identified elements such as ontology, roles and interactions, are refined. At the end of the design phase, the hierarchical organisation structure is mapped to a holarchy (hierarchy of holons) in charge of its execution. Each of the previously identified organizations is instantiated in forms of groups. Corresponding roles are then associated to holons or agents. This last activity also aims at describing the various rules that govern the decision making inside composed holons as well as the holons' dynamics in the system. All of these elements are finally merged to obtain the complete set of holons (composed or not) involved in the solution. In this way, the complete holarchy of the solution is described.

The *Implementation* and *Deployment* phase aims at implementing the agent-oriented solution designed in the previous phase by adapting it to the chosen implementation platform (in our case: the Janus platform [15]). The implementation phase details activities that allow the description of the solution architecture and the production of associated source code and tests. It also deals with the reuse of previously developed solutions. The *Deployment* starts with the description of the deployment configuration; it details how the previously designed application will be concretely deployed; this includes studying distribution aspects, holons physical location(s) and their relationships with external devices and resources. Finally, test activities are performed.

# 4 The concept of Capacity through the development process

The concept of capacity is one of the key concepts of the CRIO metamodel underlying the ASPECS development process. This section describes the various faces of this concept mainly during the two first phases of the ASPECS development process: the system requirements analysis (subsection 4.1) and the agent society design (subsection 4.2). It also describes the way a holon may dynamically acquire a new capacity at runtime (subsection 4.3).

## 4.1 The concept of Capacity during the analysis phase

A Capacity describes what a behavior is able to do or what a behavior may require to be defined. As a consequence, there are two main ways of using this concept: (i) It specifies the result of some role interactions, and consequently it specifies results that an organisation as a whole may achieve with its behavior. In this sense, it is possible to say that an organisation may exhibit a capacity. (ii) Capacities may be used to decompose complex role behaviours by abstracting and externalizing (for instance by delegating to other roles) a part of their role tasks into capacities. In this case the capacity may be considered as a behavioral building block that increases modularity and reusability of roles and organisations. Thanks to this dual aspect, capacity allows to make an interface between two adjacent levels of abstraction in the organizational hierarchy of the system. In this context, capacity may be used to specify the contributions among already identified organisations located at different levels of abstractions (level $n$ and $n-1$). It may also be used as a new requirement to identify new organisations at a lower level of abstraction (level $n-1$) and thus break down the complexity of a role (level $n$). A role at level $n$ requires a capacity that is in turn provided by an organisation at level $n-1$.

To better illustrate the dual aspect of the capacity concept, an example will be provided: the capacity to find the shortest path in a weighted directed acyclic graph $\mathcal{G}(N, E)$, from a source node to a destination node $d$. This capacity is formally described in figure 3. The template includes the *name* of the capacity, its required *inputs* and what it produces as an *output*. Constraints on the properties of inputs and outputs are defined in the *requires* and *ensures* slots respectively. A *textual description* gives an informal description of the capacity.

This capacity may be realized in various ways. *Dijkstra* [10] or *Bellman-Ford* [4] algorithms may be used if the know-how of a single entity is considered. Besides other realizations may be found, especially if the know-how of a group of entities modeled by an organisation is considered. The *Ant Colony* is a well-known organisation able to find a solution to the problem of finding the shortest path in a graph [3]. The solution (the shortest path) emerges from interactions among Ants in their environment. Let us suppose that the environment represents the graph $\mathcal{G}$, the source node $s$ is mapped to the Ant Hill and the destination $d$ to a food source. Figure 4 represents the design of a portion

Name: FindShortestPath
Inputs: $\mathcal{G} = (N, E)$, directed graph. $E = N \times N$
$w : E \to \mathbb{R}$, weight function.
$s \in N$, source node.
$d \in N$, destination node.
Outputs: $P = \langle s = i_0, i_1, \cdots, i_{n-1}, d = i_n \rangle$,
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.

Figure 3: Formal description of FindShortestPath capacity

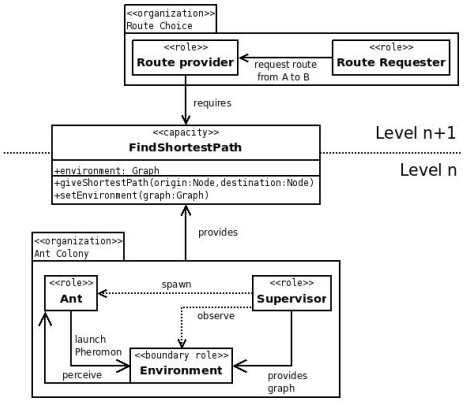


Figure 4: The concept of capacity as a link between two adjacent levels of abstraction during the analysis phase

of a system composed by several levels of abstractions. At the level $n+1$, the *Route Choice* organisation is responsible for providing the best route between two given points to another organisation not represented in the diagram (for instance the *Motion Control* organisation responsible to control the movement of a robot). The request of finding the route is done by the *Route Requester* role (probably played by a member of the *Motion Control* organisation) responsible to obtain the required information. The route is chosen by the *Route Provider* role that is, indeed, not able to do that by itself, this latter requires the *Find-ShortestPath* capacity that actually provides the result. This capacity provides the solution of a problem that is effectively solved at a lower level of abstraction (level $n$). Figure 4 proposes one possible implementation to this capacity by means of an ant colony thanks to the *Ant Colony* organisation that is located at a lower level in the organizational hierarchy. The capacity concept thus allows to define how an organisation at level $n$ may contribute to the behavior of a role at level $n+1$.

Let us consider the need of modeling a complex system behavior. It is assumed that it is possible to decompose the system from a functional point of view into a set of finer grained (less complex) behaviours interacting to meet the objectives of the organisation. Depending on the considered level of abstraction, an organisation can be seen either as a single behavior or as a set of interacting behaviours. The concept of organisation is inherently a recursive one [13]. The same duality is also present in the concept of holon. Both are often illustrated by the same analogy: the composition of the human body. The human body, from a certain point of view, can be seen as a single entity with an identity and its own behavior and personal emotions. Besides, it may also be regarded as a cluster/aggregate of organs, which are themselves made up of cells, and so on. At each level of this nested hierarchy, specific behaviours emerge [6]. The body has an identity and a behavior that is unique for each individual. Each organ has a specific mission: filtration for kidneys, extraction of oxygen for lungs or blood circulation for the heart, etc. An organisation is either an aggregation of interacting behaviours, and a behavior composing an organisation at an upper level of abstraction; the resulting whole constitutes a hierarchy of behaviours that has specific goals to be met at each level.

This recursive definition of the organisation will form the basis of the analysis activities performed within the ASPECS development process. The system global behavior is recursively decomposed into a set of interacting sub-behaviours, each of these latter being in turn decomposed until the lowest level of elementary sub-behaviours is reached. It means that at a given level, composed behaviours are modeled by using organisations. These organisations are composed of roles which can be in turn decomposed into organisations at a lower level. The concept of capacity is used to specify contributions between the various organisations located at different levels of abstraction. In most systems, it is somewhat arbitrary as to where the partitioning is left off and what subsystems are taken as elementary (cf. [35, chap. 8]). This choice remains a pure design choice. During the design phase, the hierarchical organisation structure resulting from the analysis will be mapped to a holarchy (hierarchy of holons) in charge of

its execution. Each of the previously identified organisations is instantiated in forms of groups. Corresponding roles are then associated to holons to obtain a holarchy able to execute the various behaviours identified during the analysis.

## 4.2 The concept of Capacity in the design phase

As already discussed, during the agent society design phase, the capacity concept is also used as an interface between organisations located at different levels of abstraction, as described in figure 5. But these inter-level contributions are refined by the introduction of the concept of service provided by an organisation and implementing the required capacities.

A service implements a capacity thus accomplishing a set of functionalities on behalf of its owner, a role. This definition of service is inspired from the Web Services Architecture [40] of the W3C[2]). These functionalities can be effectively implemented by a set of capacities required by the owner role. A role can thus publish several of its capacities and other members of the group can take profit of them by means of a service exchange. Similarly a group, that is able to provide a collective capacity can share it with other groups by providing a service.

The relationship between capacity and service is thus crucial in our meta-model. A capacity is an internal aspect of an organisation or a role, while the service is designed to be shared among various organisation or entities. A service is created to publish a capacity and thus to allow other entities to benefit from it.

It will be assumed that an organisation is able to provide a service if the service is provided by one of its roles. A role may provide a service if it manages/supervises a workflow where either:

- the service is implemented in one of its own behaviours; in this case, obviously, the role exhibits a capacity with a compatible description.
- the service is obtained from a subset of other roles of the same organisation by interacting with them using a known protocol; this is a service composition scenario where the work plan is a priori known and the performance level may be someway ensured.
- the service is obtained from the interaction of roles of the same organisation but the work plan is not a priori known and the service results in an emergent way (i.e., Ant Colony Organisation).

In the last case, the management of a service provided by the global organisation behavior is usually associated to one of the organisation's roles (i.e., Supervisor in the Ant Colony Organisation). It may also be managed by at least one of the representative of the various holons playing roles defined in the corresponding organisation.

If we consider again the example of the *FindShortestPath* capacity, the Ant Colony organisation is able to provide this capacity. This latter is then required

[2]World Wide Web Consortium: `http://www.w3.org`

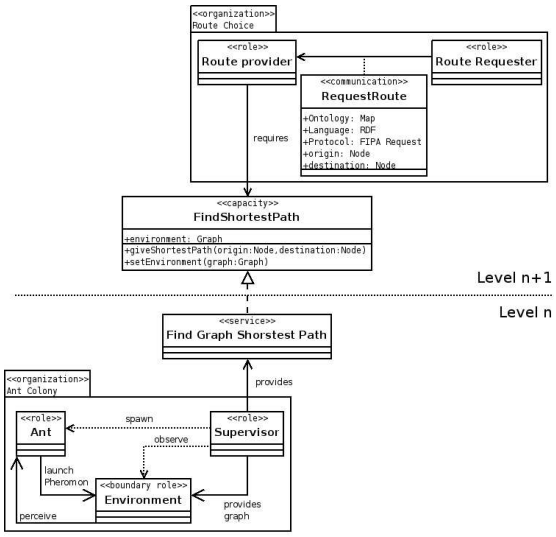


Figure 5: The concept of capacity during the design phase

to define the behavior of Route Provider role (see figure 5). The organisations are located at two different levels of abstraction.

As shown in Figure 6, the holonic approach enables to simply represent both levels by creating a super-holon whose members are ants. The two previously discussed organisations are instantiated in form of groups $g_0$ and $g_1$. The group $g_0$ is composed of three members $H_1$, $H_2$ and $H_3$ (level $n$). The holon $H_3$ plays the *Route Requester* role; it wishes to obtain the shortest route to its destination, and for that, it queries holon $H_1$ acting as *Route Provider*. The behavior of this latter role is based on the *FindShortestPath* capacity. The holon $H_1$ must have a realization of that capacity. Suppose for example he opts for the implementation based on the *Ant Colony* organisation. In this case, the $H_1$ holon will have an instance of that organisation, noted *g2: Ant Colony*, see figure 6. Despite the fact that the Ant Colony is providing a specific service, none of its members (members of g2) is able to play the *Route Provider* role by itself. The super-holon $H_1$ is thus able to play a role that is inaccessible to its members.

Agent and holon concepts are introduced during the design phase. In this way, the concept of capacity may be used in a new manner, as an interface between an agent and a role.This is typically the case for capacities identified in the lowest level of the system organizational hierarchy. To obtain a generic model of organisation, it is necessary to define a role without making any assumptions on the architecture of the holon which will play this role. Basing the description of the role behaviours 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, while its implementations are the way to exploit that competence.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.

During design phase, a super-holon can obtain a capacity, either by directly getting a concrete implementation in the form of algorithms (i.e., *Dijkstra* or *Bellman-Ford*), or by incorporating a group (of production) capable of providing a service that achieves the capacity it needs. This last case is really interesting since it may be used at runtime to dynamically change the capacity implementations of a given holon. The next section details the process of dynamic capacity acquisition.

### 4.3 The concept of Capacity in the deployment phase: dynamic capacity acquisition

Implementation and Deployment phase is based on the Janus platform that was specifically designed to deal with the implementation and deployment of holonic and multiagent systems [15]. It is based on an organizational approach and its key focus is that it supports the implementation of the concepts of capacity, role and organisation as first-class entities.

One of the fundamental advantages of the concept of capacity is that it allows an agent/holon to reason about its own skills and those of other agents/holons. In Janus, each holon has a set of basic capacities that may dynamically evolve. In order to acquire a new capacity a holon may instantiate a new internal organisation providing a service that implements the required capacity. A possible scenario for the exploitation of this interesting feature is related to the already introduced *FindShortestPath* capacity example. Let us initially suppose that the group $g_0$ of figure 6 contained only two members: a *Route Requester* role (played by holon 3) and only one *Route Provider* role (played by holon 2). The holon 1 wishes to improve the capability of this group by playing the *Route Provider* by itself (in this way two holons in the group can offer the same service) but it does not possess the required capacity (*FindShortestPath*). The holon 1 has thus two main possibilities for accomplishing its goal. The first possibility consists in the recruitment of a member, already owning the *FindShortestPath* capacity. The second possibility consists in instantiating an organisation able to provide it, like the *Ant Colony* organisation. This organisation is found by using a matchmaking process. Let us consider the situation where holon 1 chooses this second option and integrates an additional internal group, thus instantiating the *Ant Colony* organisation as described in figure 6 by an extension of the cheese-board diagrams introduced in [12]. It recruits new members able to play the various role of this organisation (cf. fig 5). Owning henceforth the required capacity, it becomes able to play the *Route Provider* role and thus joins the group $g_0$.

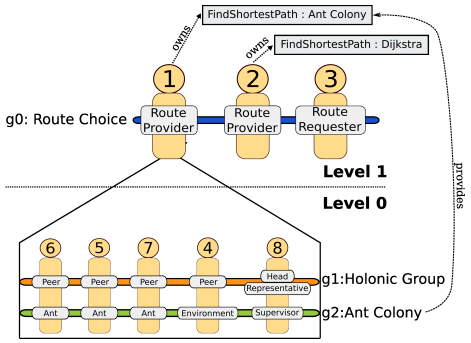


Figure 6: Acquiring a new capacity by integrating a new internal organisation

This process can be summarized by the following steps :

1. The holon tries to match the capacities required to play one role as it is stated by the organisation description. If it cannot then it looks for an organisation exhibiting the needed capacity (in an organisations repository). To enable this matchmaking process in an open system, a common description language is required to match holon capacities, as proposed in [37, 38] 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 organisations the one which seems the best fitting. This choice is mainly based on the comparison between capacities required by the chosen organisation and the already present members' capacities.

3. When an organisation has been chosen, the holon has yet to instantiate the defined roles and interactions. Roles can be played by existing sub-holon members or new members capable of playing those roles have to be recruited.

4. When each role defined in the chosen organisation has a player in the freshly instantiated group, the super-holon is able to obtain the capacity implementation and can thus play new roles.

The following section introduces an example in order to illustrate the use of the capacity concepts in the ASPECS methodology.

# 5 Case study: a robot soccer simulator

The FIRA Robot soccer competitions began in 1996 using real robots and simulators [21]. It is an example where real-time coordination is needed. Indeed, the principle consists in two teams of five autonomous robots (for the specific

case of Mirosot medium league) that play a game similar to human football. It constitutes a well-known benchmark for several research fields, such as MAS, image processing and control. A simulator for such games based upon HMAS using the ASPECS development process was developed. In this paper some parts of the ASPECS process concerning this simulator are presented. Not all activities are detailed since this would be out of the scope of this paper. The focus here is on pertinent activities that allow to engineer the emergence of the desired behavior. The interested reader can find the entire case study on the ASPECS website[3].

## 5.1 System Requirements

The global objective of the Domain Requirements Description (DRD) activity is gathering requirements and expectations of application stakeholders and providing a complete description of the behavior of the application to be developed in terms of functional and non-functional requirements described using an UML use case diagram. Figure 7 details the use cases associated to the development of a simulator for the FIRA Robot Soccer cup. Eight use cases and one actor have been identified. The actor represents the user of the simulator who can simulate matches and tune the strategy of each team. Simulating a match implies the simulation of two autonomous teams that can choose their own strategy and are responsible for simulating the individual robots behavior.

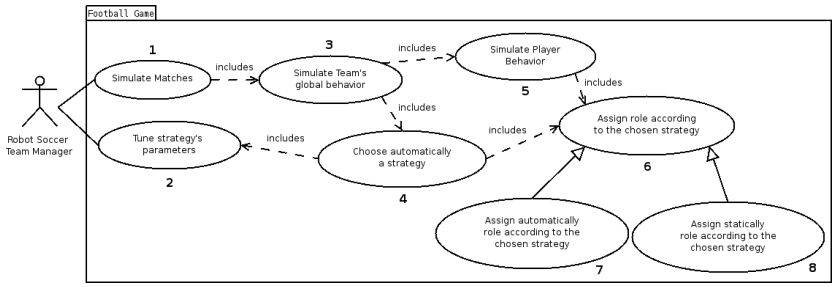


Figure 7: Domain Requirements Description of the Robot Soccer Simulator Analysis

The goal of the Organisation Identification activity (the second one in ASPECS) is to bind each requirement to a global behavior, embodied by an organisation. Starting from the results of the DRD activity, use cases are clustered and a first set of organisations that will compose the application is identified by following a combination between a structural (or ontological) approach mainly based on the analysis of the problem structure described in the Problem Ontology Description and a functional approach based on requirement clustering.

[3] http://www.aspecs.org/

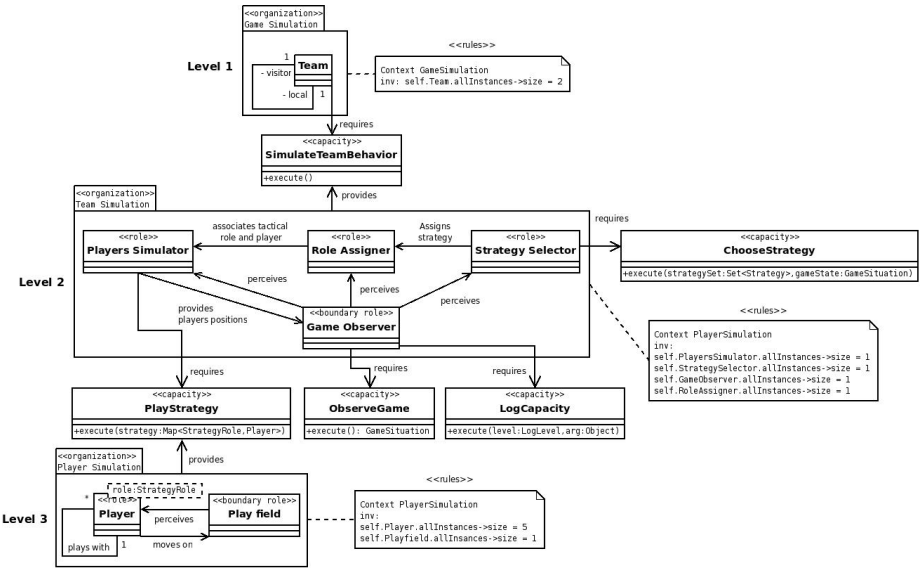


Figure 8: Results of Interaction and Role Identification, and Capacity Identification activities of the Robot Soccer Simulator Analysis

This step relies on analysis decision. In the reported example, a functional criterion is used and organisations are clustered according to use dependencies, specifically includes relationships. For example, the top level organisation consists in game simulation. This organisation is dedicated to the realization of the *Simulate Matches* use case and it is composed of The *Team Simulation* organisation which is dedicated to simulate team's global behavior use case.

According to this first hierarchy of organisations, the objective is now to decompose each organisation in terms of roles and interactions, and precise the behavioral contributions of sub-organisations to upper-level ones. In this example a top-down behavioral decomposition is used and for each level a test is done to determine if it is still necessary to continue the decomposition process. Figure 8 describes the result of this decomposition: the various organisations and their respective contributions.

At the top of the hierarchy, the *Game Simulation* organisation is decomposed using only one interaction and one role: *Team*. An OCL constraint is added to specify that only two instances of this role are allowed in each instance of this organisation. This role is in charge of simulating the behavior of a robot soccer team. Its complexity at this level is considered as too high for a straight implementation and it will be decomposed into smaller interacting behaviours.

At the second level, the organisation *Team Simulation* is decomposed in four roles: three roles representing the contributions of previously identified sub-

level organisations and a boundary role [4] useful to provide required information about the game situation (players and ball position, score, etc). The *Strategy Selector* role is in charge of determining the best strategy to adopt according to the current situation. Each strategy corresponds to a distribution of strategy roles like *goalkeeper*, *near-defender*, *midfielder*, *shooter*, etc., among the different players. Each robot soccer team is composed of five players. Applying a strategy thus consists in assigning to each player one of the roles defined by the strategy. The association between players and roles defined by the strategy is done by the *Role Assigner* role. The *Players Simulator* role is in charge of simulating the behavior of the various players according to the chosen strategy. The global state of the game (players and ball positions, score, time, etc.) is maintained by the boundary role *Game Observer*. This role is also in charge of providing required perceptions to the others.

The result of the Interaction and Role Identification activity is thus a three levels holarchy composed of the organisations depicted in figure 8. The analysis is obviously not finished and many decisions have to be taken before being able to implement and deploy the system. The behaviours of the roles need to be defined. The *Strategy Selector* role, for example, is in charge of choosing a strategy for a team according to a game situation. This role is of an uttermost importance for the team efficiency and the choice of a strategy is not so trivial. However there exist several techniques in order to exhibit such a behavior.

We decided to abstract the choice of a strategy with a capacity namely *ChooseStrategy* capacity. This strategy inputs are a game situation and a non empty set of strategies; it returns the selected strategy as a result. At this step, this capacity may be considered as a new requirement. The System Requirements Analysis phase need to be reiterated to consider this new requirement and thus identify organisations in charge of realizing it. The next subsection presents such an iteration which results in an organisation based upon the theory of immune systems that is able to implement the *ChooseStrategy* capacity.

## 5.2 The Immune System Organisation

From a computational viewpoint the human immune system can be viewed as a parallel, distributed system that has the capacity to control a complex system over time [11]. The human immune system is composed of several layers of defense such as: physical (skin), innate and adaptive. The adaptive part of the human immune system has been mostly taken into consideration in this paper. The adaptive system improves its response to a specific pathogen at each exposure. Thus, the adaptive system has three key functionalities: recognition, adaptation and memory. The adaptive immune system can be divided into two major sections: the humoral immune system and the cellular immune system. The former acts against antigens by means of proteins called immunoglobulins or antibodies which bind to antigen. This binding mechanism allows an antibody to

[4] A Boundary Role is a role located at the boundary between the system and its outside, and it is responsible for interactions happening at this border (i.e. GUI, Database, etc).

either tag an antigen for attack by other part of the immune system or neutralize antigens. The latter, among other duties, destroys virus-infected cells.

Nobel Laureate N. K. Jerne proposed a model based on interactions between antibodies [20] to explain the human immune systems functionning. These interactions take the form of stimulation and inhibition. This theory is known as Jerne's Idiotypic Network. The network is defined by stimulation/inhibition links between antibodies. From now on, immune system term will be used as a reference to the immune network. The region by which antibodies stimulate or inhibit other antibodies is called idiotope. Idiotopes play the roles of antigens for other antibodies. It means that each antibody may be seen by other antibodies as an antigen if its idiotope corresponds to the paratope of these antibodies. This regulation mechanism enables the immune system to maintain an effective set of cells and to self-organize in order to deal with antigens. Indeed, the stimulation/inhibition links are based on affinities between antibodies to deal with specific antigens. If two types of antibodies are able to match two similar type of antigens then they will have affinity and will stimulate each other. On the contrary if antibodies are built to deal with very different types of antigens they will inhibit themselves.

Jerne's Idiotypic Network has already been used as an agent architecture, for example in [41]. Such architecture is an interpretation of the Jerne's theory. Concepts developed in that work are used for a single agent case as a basis for the approach presented in this paper. The main principle of this architecture is that each antibody represents a possible behavior of the agent with its preconditions and affinities with other antibodies. It is an arbitration mechanism which allows both the choice of a single behavior according to some stimulations, the antigens, and learning with the continuous computation of affinities between these antibodies. In our case the behavior will be associated to a strategy and the choice of a behavior is then equivalent to the choice of a strategy.

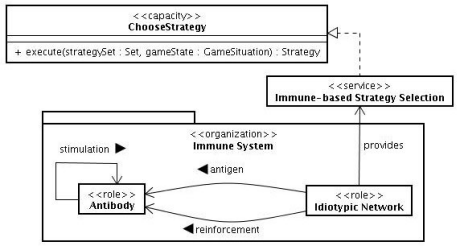


Figure 9: Organizational model of the Artificial Immune System

Figure 9 describes the organizational model of the immune system. It is composed of two roles: *Idiotypic Network* and *Antibody*. The Antibody role describes the behavior exhibited by *Antibodies* in the network. *Antibodies* influence each other through the *affinities interaction*. When appropriate, *antibodies* will send stimulation/inhibition stimuli, according to the affinities values of

the *antibody*, to other *antibodies*. With these stimulations/inhibitions and the present antigens, each *antibody* computes its concentration. The *antibodies* then send their concentrations to the *Idiotypic Network* for selection. The *antibody* with the greater concentration is chosen and executes its behavior. After execution the results are analyzed by the *Idiotypic Network* which sends rewards or penalties to antibodies in order to update their affinities.

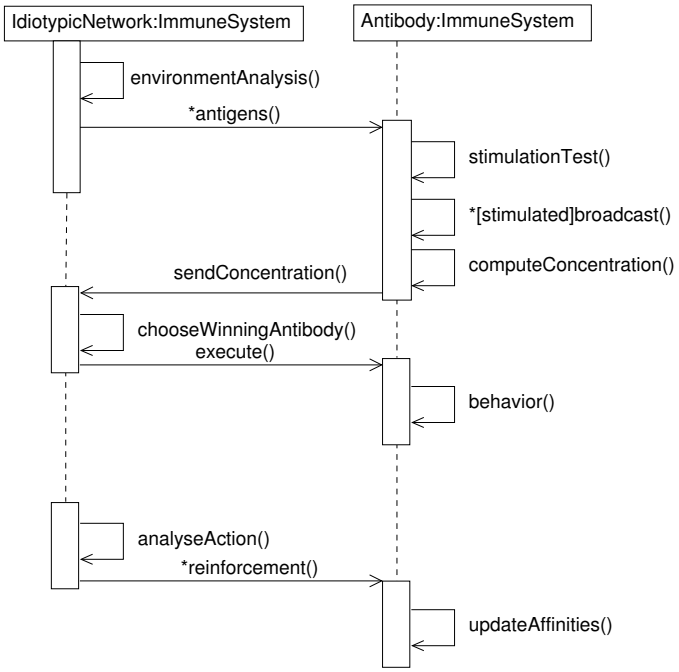


Figure 10: A complete step of the Immune System interactions

Figure 10 details the interactions resulting from the perception of an antigen by the immune system. The perception and encoding of antigens is done by an analysis of the environment. The perceived antigens are sent to all antibodies. Each antibody checks if it is stimulated by the antigen and if stimulated it broadcasts its affinity values. It means that a stimulated antibody will either stimulates or inhibits other antibodies.

More details on this agent architecture and experiments can be found in [19].

## 5.3 The Agent Society Design Phase

The global objective of the ASPECS Agent Society phase is to define roles' communications, agents' and holons' architectures according to the organisations identified in the System Requirements phase. The activities presented in this section are those that are pertinent to the engineering of the emergent feature

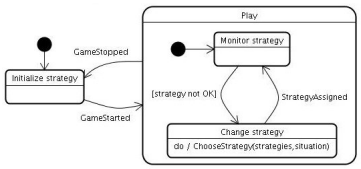


Figure 11: Behavior of Strategy Selector role

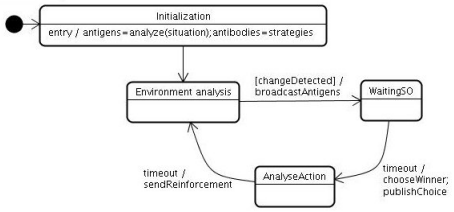


Figure 12: Behavior of Idiotypic Network role

exemplified in this paper. These activities are: Role Behavior Description, Organisation Dependencies and Holarchy Design.

The Role Behavior Description activity aims at defining the complete lifecycle of a role. Roles identified during the IRI activity are here specialised in Agent Roles interacting each others by means of communications. If a role requires capacities or provides services, this activity has to describe tasks and actions in which they are really used or provided. This is shown in figure 11 by the statechart representing the behavior of the Strategy Selector role. The Change strategy state calls the *ChooseStrategy* capacity. This capacity as analyzed in the System Requirements phase is realized by an artificial immune system inspired by the idiotypic network theory. The behavior of a role of the corresponding organisation is defined by the statechart of figure 12. This role, named Idiotypic network, consists in coordinating antibodies and simulating the global behavior of an idiotypic network. In our case, the antigens are the result of the analysis of the game situation. The different strategies are represented by antibodies. These two inputs are given to the capacity call in the ChangeStrategy state of role Strategy Selector. The Idiotypic Network role is then in charge of choosing among these antibodies, strategies in our case, one which seems the best fitting for the current situation. Once an antibody is chosen, the corresponding result is published as a provided service.

The goal of the Organisation Dependencies Description activity is to define relationships between: (i) capacities required by roles and organisations, and (ii) services that realize them. It is also dedicated to the identification of resources but this part will not be detailed here for the sake of brevity.

It is the case in our example since the Idiotypic Network role exhibits the

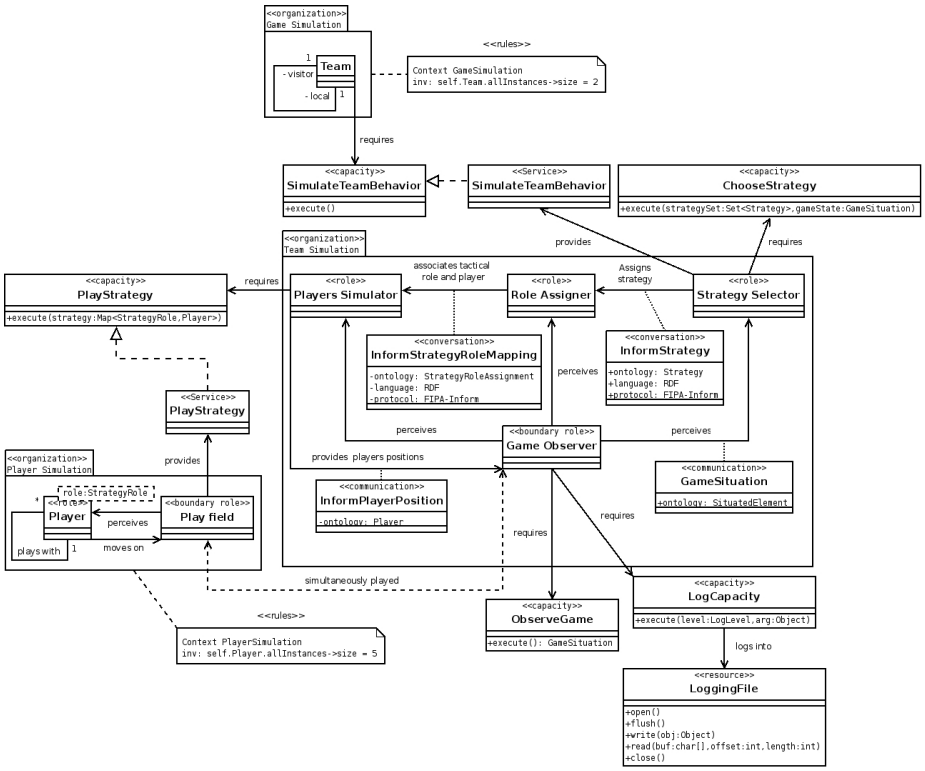


Figure 13: Organisation Dependencies Description of the Team Simulation Organisation

result of the Immune System organisation.

Figure 13 describes capacities, services and resources related to the three organisations involved in the model of the Robot Soccer Simulator case study. Contributions of each organisation to the upper level have been detailed by using a capacity and the associated service, e.g., *SimulateTeamBehaviour* or *PlayStrategy*.

The Holarchy Design activity is the last activity of the Agent Society design phase and aims at providing a global synthesis where previous activities work products are combined and summarized in a single work-product describing the overall structure of the system and the rules that will govern the dynamics of this structure. At this step in the design process, the set of organisations composing the system, their roles and the associated communications have been identified and specified. The architectures of the various agents have also been specified.

Finally the designer focuses on the Holarchy Definition task. The previously described elements are merged in order to obtain the complete set of holons (composed or not) involved in the solution. In this way, the complete holarchy

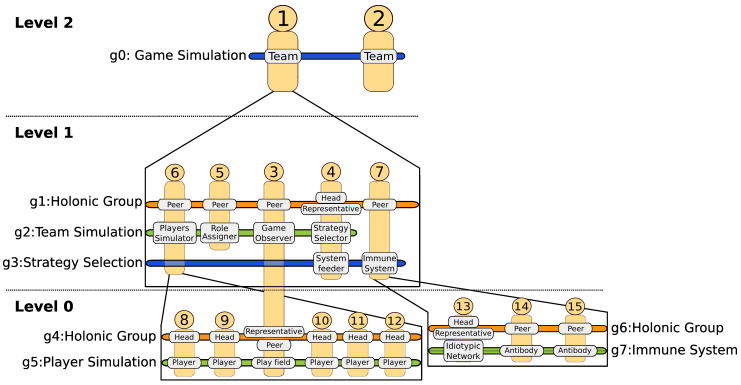


Figure 14: Overview of the resulting holarchy

of the system is described. Results of this task are summarized in an organizational cheese-board (see Figure 14). This diagram is then associated to a set of documents describing the government of each holon and the rules governing their dynamics.

Figure 14 shows that the StrategySelector role is in a group with an Immune System role. In this role the Strategy Selector is playing the System feeder role. The Immune System is in turn decomposed in a group where there exist one Idiotypic Network and several Antibodies according to the immune system organizational structure presented in section 5.2.

# 6 Conclusion

Engineering emergent behaviours is an open issue in self-organisation even if some works propose some interesting ideas [26, 27, 36, 9, 32]. The same problem also affects the engineering of holonic multi-agent systems since any level of a holarchy can depend on the emergent behaviours of its sub-levels. In order to tackle this specific feature of holonic and self-organizing multi-agent systems, the capacity concept which abstracts a know-how from its concrete realisation is introduced in this paper. The use of this concept is illustrated in this paper through a case study demonstrating the use of the capacity concept to control the emergence of behaviours. The chosen case study consists in a simulator for robot soccer games and it is considered under a holonic perspective. The capacity concept is part of the metamodel underlying the ASPECS development process. The entire set of activities of this process is out of the scope of this paper so only relevant activities applied to the case study are presented. This paper emphasizes the intrinsic ability of ASPECS to catch the various levels of abstraction of a complex system; this occurs during the analysis phase by using an organisational hierarchy and in the design phase by using an holarchy.
The obvious hypothesis of the presented works is the existence of an holarchy.

It means that entities can be decomposed in a somewhat hierarchical manner. This hypothesis is not valid for flat systems but in the case of emerging properties authors frequently refer to, at least, two levels. The lower level is where entities interact to produce the emergent phenomenon and the upper level is the one in which the phenomenon is observed. A second hypothesis made in this work is that analysis, design and development use an organisational stance. Again, this hypothesis seems reasonable as many methodologies, [43, 5, 7] for example, use similar concepts.
Future works will be concerned with the language for the expression of complex capacities. As it is now it seems appropriate for simple transformational properties. Validation and verification of these properties will be integrated within the methodology to form a complete lifecyle.

# 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, *CEEMAS 2005*, number 3690 in LNAI, pages 539–542, Budapest, Hungary, September 2005. Springer-Verlag.

[3] Nattapat Attiratanasunthron and Jittat Fakcharoenphol. A running time analysis of an ant colony optimization algorithm for shortest paths in directed acyclic graphs. *Inf. Process. Lett.*, 105(3):88–92, 2008.

[4] Richard Bellman. On a routing problem. *Quarterly of Applied Mathematics*, 16(1):87–90, 1958.

[5] P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos, and A. Perini. TROPOS: An Agent-Oriented Software Development Methodology. *Journal of Autonomous Agents and Multi-Agent Systems*, 8(3):203–236, May 2004.

[6] Gilbert Chauvet. *La vie dans la matière*. Flammarion, 1998.

[7] Massimo Cossentino. From Requirements to Code with the PASSI Methodology. In B. Henderson-Sellers and P. Giorgini, editors, *Agent-Oriented Methodologies*, chapter IV, pages 79–106. Idea Group Publishing, Hershey, PA, USA, 2005.

[8] Massimo Cossentino, Stéphane Galland, Nicolas Gaud, Vincent Hilaire, and Abderrafia Koukam. How to control emergence of behaviours in a holarchy. In *Self-Adaptation for Robustness and Cooperation in Holonic Multi-Agent Systems (SARC), Workshop of the second IEEE International*

*Conference on Self-Adaptive and Self-Organizing Systems (SASO)*, Isola di San Servolo, Venice, Italy, October 2008.

[9] J.L Dessalles, J.P. Mller, and D. Phan. Emergence in multi-agent systems: conceptual and methodological issues. In F. Amblard and D. Phan, editors, *Multi- Agent Models and Simulation for Social and Human Sciences*. Oxford, The Bardwell-Press, 2007.

[10] E.W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, December 1959.

[11] J. Doyne Farmer, Norman H. Packard, and Alan S. Perelson. The immune system, adaption and machine learning. *Physica D*, 22:187–204, 1986.

[12] J. Ferber, O. Gutknecht, and F. Michel. From Agents to Organizations: an Organizational View of Multi-Agent Systems. In *AOSE-IV@AAMAS03*, volume 2935 of *LNCS*, pages 214–230. Springer Verlag, mar 2004.

[13] Jacques Ferber. *Multi-Agent Systems. An Introduction to Distributed Artificial Intelligence*. Addison Wesley, London, 1999.

[14] Stephanie Forrest. Emergent computation: Self-organizing, collective, and cooperative phenomena in natural and artificial computing networks. *Physica D*, 42:1–11, 1990. (Proceedings of the Ninth Annual Int. Conf. of the Center for Nonlinear Studies, Ed. Stephanie Forrest).

[15] Nicolas Gaud, Stéphane Galland, Vincent Hilaire, and Abderrafiâa Koukam. An Organisational Platform for Holonic and Multiagent Systems. In *PROMAS-6@AAMAS'08*, Estoril, Portugal, May 12-16th 2008.

[16] C. Gerber, J.H. Siekmann, and G. Vierke. Holonic multi-agent systems. Technical Report DFKI-RR-99-03, DFKI - GmbH, 1999.

[17] P. Gruer, V. Hilaire, A. 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.

[18] 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, *ESAW*, number 1972 in LNAI. Springer Verlag, 2000.

[19] Vincent Hilaire, Abder Koukam, and Sebastian Rodriguez. An adaptative agent architecture for holonic multi-agent systems. *ACM Transactions on Autonomous and Adaptive Systems*, 3(1):1–24, 2008.

[20] N. K. Jerne. Towards a network theory of the immune system. *Ann Immunol (Inst Pasteur)*, 125C:373–389, 1974.

[21] Yung Ho Kim. *Micro-robot world cup soccer tournament*. KAIST, 1996.

[22] Arthur Koestler. *The Ghost in the Machine*. Hutchinson, 1967.

[23] Eric Matson and Scott A. DeLoach. Autonomous organization-based adaptive information systems. In *KIMAS '05*, Waltham, MA, April 2005.

[24] 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.

[25] Francisco Maturana. *MetaMorph: an adaptive multi-agent architecture for advanced manufacturing systems*. PhD thesis, The University of Calgary, 1997.

[26] H. Van Dyke Parunak and Sven A. Brueckner. *ENGINEERING SWARMING SYSTEMS*, pages 341–376. Kluwer academic publisher, 2004.

[27] Van Dyke Parunak. "go to the ant": Engineering principles from natural multi-agent systems. In *Engineering Principles from Natural Agent Systems. Annals of Operation Research*, 1997.

[28] Loris Penserini, Anna Perini, Angelo Susi, and John Mylopoulos. From capability specifications to code for multi-agent software. In *ASE*, pages 253–256. IEEE Computer Society, 2006.

[29] S. Rodriguez, N. Gaud, V. Hilaire, S. Galland, and A. Koukam. An analysis and design concept for self-organization in holonic multi-agent systems. In S. Bruckner, S. Hassas, M. Jelasity, and D. Yamins, editors, *Engineering Self-Organising Systems*, volume 4335 of *LNAI*, pages 15–27. Springer-Verlag, 2007.

[30] J.C. Routier, P. Mathieu, and Y. Secq. Dynamic skill learning: A support to agent evolution. In *AISB'01*, pages 25–32, 2001.

[31] 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.

[32] Michael Schillo, Bettina Fley, Michael Florian, Frank Hillebrandt, and Daniela Hinck. Self-organization in multiagent systems: From agent interaction to agent organization. In *Proceedings of the Third International Workshop on Modelling Artificial Societies and Hybrid Organizations (MASHO)*, August 20 2002.

[33] G Di Marzo Serugendo, N Foukia, S Hassas, A Karageorgos, S Kouadri Mostefaoui, O F Rana, M, P Valckenaers, and C Van Aart. Self-organisation: Paradigms and applications. In *Proceedings of ESOA'03*, 2003.

[34] Giovanna Di Marzo Serugendo, Marie-Pierre Gleizes, and Anthony Karageorgos. Self-organisation and emergence in mas: An overview. *Informatica*, 30(1):45–54, 2006.

[35] Herbert A. Simon. *The Science of Artificial*. MIT Press, Cambridge, Massachusetts, 3rd edition, 1996.

[36] Susan Stepney, Fiona Polack, and Heather R. Turner. Engineering emergence. In *ICECCS*, pages 89–97. IEEE Computer Society, 2006.

[37] 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.

[38] 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.

[39] M. Ulieru and A. Geras. Emergent holarchies for e-health applications: a case in glaucoma diagnosis. In *IEEE IECON 02*, volume 4, pages 2957–2961, 2002.

[40] W3C. *Web Services Architecture*. World Wide Web Consortium, February 2004.

[41] Y. Watanabe, A. Ishiguro, and Y. Uchkawa. Decentralized behavior arbitration mechanism for autonomous mobile robot using immune system. *Books Artificial Immune Systems and Their Applications, Springer-Verlag, p. 186-208, ISBN 3-540-64390-7*, 1999.

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

[43] F. Zambonelli, N. Jennings, and M. Wooldridge. Developing multiagent systems: the GAIA methodology. *ACM Trans. on Software Engineering and Methodology*, 12(3), 2003.