Metrics for Evaluating Modularity and Extensibility in HMAS Systems

Massimo Cossentino, Carmelo Lodato, Salvatore Lopes, Patrizia Ribino
National Research Council of Italy - I CAR
cossentino,c.lodato,s.lopes,ribino@pa.icar.cnr.it

Valeria Seidita*
DICGIM
University of Palermo - Italy
valeria.seidita@unipa.it

ABSTRACT

Nowadays, software systems are more and more frequently designed in order to realize complex dynamical behavior for solving complicated problems. Holonic Multi Agent Systems (HAMAS) is spreading for the development of such systems since they allow to manage system requirements in terms of behaviors and organizational patterns. Traditional software engineering metrics are not useful for measuring HMAS architectures since they do not consider different nested levels of organizational structures. We want to contribute to this issue proposing some metrics for evaluating modularity and extensibility of HMAS architectures.

Categories and Subject Descriptors

D.2.8 [Software Engineering]: Metrics—complexity measures

General Terms

Measurement

Keywords

Metrics; Holonic Multi Agent System; Holon

1. INTRODUCTION

In several software engineering paradigms, for instance the object-oriented one, designers may rely upon strengthened metrics for evaluating and assessing, in a quite early stage of design, the quality of the architecture they are using; indeed, literature provides metrics for several quality attributes [8][4][21].

The same situation is not present in Agent Oriented Software Engineering (AOSE) and in particular in one of the most promising AOSE paradigm for facing problems related to modeling and developing complex systems, that is the Holonic Multi Agent System (HAMAS) paradigm. Multi Agent Systems (MASs) have been often used as a software paradigm for developing complex systems due to the particular features of their components. The increasing complexity of the systems in terms of behaviors and organizational patterns makes the multi agent design paradigm unable to manage alone different nested levels of organizational structures. In recent years, HMAS is spreading for the development of complex systems. But, because of the wide range of holonic architectures, in the same way of other SE areas, an open question is: how can we choose the most useful one? Which are the quality attributes that may be useful for evaluating HMAS architectures?

In [12] Magariño et al. defined a set of metrics with the aim of measuring some quality attributes of MAS architectures such as: extensibility and modularity. Extensibility indicates how much MAS architectures are suitable for answering to extensions and enhancing without heavy changes in their infrastructure. Modularity indicates whether, and at which degree, architectures (hence MAS components) may be divided and recombined without great cost.

In order to evaluate these quality attributes, [12]’s authors used a selected set of metrics coming from object oriented software engineering such as cohesion, coupling. Fan-in and Fan-out to name a few. These metrics refer to the concepts of modules, components and dependencies that are natively object oriented and Magariño et al. found the analogous concepts in the agent oriented architectures redefining object oriented metrics in order to be applied to MASs. This paper starts from this work and aims at identifying metrics for evaluating modularity and extensibility quality attributes for HMAS architectures. In this work we present a deep analysis made on the HMAS systems showing that the metrics defined for MASs are often not suitable for HMASs. Thus, we propose what are the holonic concepts corresponding to the module, component and dependency to be considered in order to evaluate extensibility and modularity of HMASs. Hence, we define the coupling and cohesion metrics for HMASs.

The paper is organized as follows: section 2 illustrates the background and motivations of our work. Section 3 shows the basic concepts of object oriented metrics, their mapping to the agent oriented one and then to the HMAS paradigm. In sections 4 and 5, the HMAS metrics are presented and then some discussions and conclusions, in section 6, are drawn.

2. BACKGROUND AND MOTIVATION

A MAS is a system composed of autonomous, reactive and proactive entities (i.e: agents) that by means of interactions allow the whole system to exhibit complex behaviors that satisfy system goals.
The design of complex systems requires the management of different levels of granularity that usually have a direct consequence in the organization design. In recent years, a new design paradigm is spreading for the development of complex systems: the holonic multi agent (HMAS) design paradigm [13][11]. Holon, which derives from the combination of holos whole and the suffix on part, is the term, coined by Arthur Koestler in his fundamental work The Ghost in the Machine [15], for indicating elements of nested hierarchies of self-replicating structures (i.e. holarchies). A holon is, commonly, defined as a self-similar structure composed of other holons as sub-structures. A useful way to implement holarchies in software system is by means of the HMAS paradigm [11]. It allows to represent a holonic system where individual agents are driven by coordination mechanism according to the cooperation rules of the holon the agent is member of. Describing a holon involves identifying and describing several organizations. An organization is used for implementing specific behaviors, tasks distribution, decision making, cooperation, etc.

In this context, it is very useful to be able to determine during the design phase, which is the most suitable holonic architecture to use in order to satisfy not only functional requirements but also some quality requirements such as having a system easily scalable and adaptable to new situations.

The main contribution of this paper is describing and measuring quality attributes for evaluating HMAS architectures.

We start our work from the one of Magariño et al. [12] that use a set of metrics coming from object oriented context for deriving metrics to measure some quality attributes of MASs. The core of that work is in finding a correspondence among the concepts of module, component and dependencies in the two contexts (OO and AO). Because of the intrinsic nature of HMAS paradigms we found the rationale Magariño et al. useful for analyzing HMAS systems features in order to retrieve the right concepts to be mapped onto the module, components and dependencies concepts for the holonic metrics.

Another work on applying software metrics for multi agent system is that of Nunes et al. [17]. In this work, authors propose the use of existing metrics such as NOA (Number of Attributes), NOM (Number of Methods) and LOCs (Lines of Code) to evaluate design modularity and stability of an evolving MAS; the work explicitly addresses features of MAS product lines and refers to concepts of Aspect Oriented Programming (AOP). From these metrics we could derive some other ones taking into account the fact that modularization is intrinsic in AOP; moreover we could map the belief and plan of BDI architecture respectively with NOA and NOM, making somehow the same work we did with Magariño et al’s metrics. This is, however, not sufficient to measure the quality of HMAS architectures. In fact, NOA, NOM, LOC and many other metrics that can be extracted from procedural program and from objects are useful for analyzing the structure of the atomic elements of the system (such as classes or agents). Other approaches consider measuring specific quality of agents, such as pro-activity or autonomy [1][2]. However they do not consider higher levels of granularity that should be considered when we deal with architectures. Conversely, the metrics defined by Magariño et al. consider specific concepts of software architectures such as module and component thus giving us a good starting point for reasoning about HMAS architectures.

3. FROM OBJECT ORIENTED TO HMAS PARADIGM

In this section, we show how we move from OO to HMAS paradigm by using the work of Magariño et al. [12] as a bridge between these two worlds. In particular, starting from the definitions of coupling and cohesion, we present an analysis made on OO concepts in order to retrieve their counterparts in the HMAS context.

Coupling and Cohesion [16][15][4] along with Fan-in and Fan-out [20][14] are steady metrics widely used in the object oriented paradigm in order to measure how strongly two or more elements depend on (or are related to) each other. In this work, we only refer to coupling and cohesion for measuring Modularity and Extensibility and for making some reasoning on the holonic multi-agent architectures. In order to do this, we need to resume the definitions of coupling, cohesion: (i) Coupling is defined as the degree of dependence between modules; (ii) Cohesion is concerned with the interactions among components within a module. Trying to adopt such metrics to the HMAS paradigm, the first step is to understand the meaning of concepts like Module, Component and Dependencies in such a context.

Module - the definition proposed by Parnas in [18] states that “a module is considered to be a responsibility assignment rather than a subprogram”. According to the same paper, a piece of a system may be considered a module if it exhibits the following properties: (i) one module may be written with little knowledge of the code in another module and (ii) one module may be reassembled and replaced without reassembly of the whole system. In the OO context a similar concept may be found for the subsystem: “sub-system is a replaceable part of the system with well-defined interfaces that encapsulates the state and behavior of its contained components” [7]. In the case of complex systems, we may recursively apply this principle and decompose a subsystem into simpler subsystems. Hence, we can affirm that subsystem may be seen as synonymous of module for our intended goals.

Components - A component is a self contained part of a module that does not verify the two properties required for a portion of a system to be considered a module. A system can be defined as a collection of components organized in a way useful for accomplishing a specific function or set of functions [6].

Dependencies - They can refer to dependencies among components of the same module (internal dependencies) and dependencies among components of different modules (external dependencies). In object oriented architectures, there are many kinds of dependencies such as one module controlling the flow of another, modules share data, one module modifies the internal content of another module and so on.

The work of Magariño et al. [12] moves from object oriented to MAS paradigm assuming agents as modules of a multi agent architecture and agent tasks (i.e. portion of agent behavior) as components. Moreover, they found similarities among OO and AO dependencies. In object-oriented architectures, some modules can share data types (i.e. data type coupling). In AO this is equivalent to several kinds of agents that share some languages referring to specific ontologies (i.e. ontology sharing). As well as, in OO some modules need to obtain data from other modules (i.e. data coupling).

In a similar way, some agents may need to request informa-
tion from other agents (i.e. knowledge coupling) and so on (see Table 1 column 1 and 2 for a complete mapping).

These similarities lead to the metrics reported in [12] for measuring the Extensibility of architectures; Extensibility is affected by the cohesion within each module of the architecture measuring the dependencies within a module, by the cohesion of the whole architecture measuring the average of the cohesion of each module and by the coupling measuring the dependencies among modules.

The cohesion within each module and the coupling are given by the following equations:

\[
C_{\text{ch}}(M) = \begin{cases} 
\frac{\text{IntDep}(M)}{\text{MaxDep}(M)} & \text{if } NC \geq 2 \\
1 & \text{if } NC = 1 
\end{cases} 
\] (1)

\[
C_{\text{cp}}(X) = \begin{cases} 
\frac{\text{ExtDep}(X)}{\text{MaxDep}(X)} & \text{if } N \geq 2 \\
1 & \text{if } N = 1 
\end{cases} 
\] (2)

M is a module, IntDep(M) is the number of the internal dependencies, MaxDep(M) = NC*(NC-1)/2 and NC is the number of components of the module; X is an agent oriented architecture, ExtDep(X) is the number of the external dependencies, MaxDep(X) is the maximum number of possible ExtDep, N is the number of modules of the architecture.

The quality attribute on Modularity may be also measured by using cohesion, coupling.

Metrics identified in [12] are the starting point for applying object oriented metrics to other fields. Different architectural styles may have specific features that influence the identification of the concepts used for defining the before said metrics. In the following section we illustrate how we derived metrics for measuring modularity and extensibility of HMAS architectures starting from this work [12].

### 3.1 Architectural Styles

An architectural style is “a set of constraints on an architecture that defines a set or family of architectures satisfying them” [3]. According to this definition, major architectural styles include several options: layers, partitions, piper and filters, client-server, three/four-tiers, model-view-controller, and so on.

According to [6, p. 3-3], a software architecture is “a description of the subsystems and components of a software system and the relationships between them”. We may regard subsystems as equivalent to the concept of module as defined by Parnas [18]. How can we map these concepts to the agent-oriented context and terminology? The easiest mapping subsystem/module we can find in an MAS is the agent itself. It naturally exhibits the two properties prescribed by Parnas and it is usually made of elements such as beliefs, rules and goals/plans that can be considered its components. Considering the agent as a subsystem/module brings a relevant consequence: software architecture overlaps agent organization. Why? According to [10]: “An organization is defined by a collection of roles that take part in systematic institutionalized patterns of interactions with other roles in a common context”. According to [6, p. 3-1]: software architecture describes “how software is decomposed and organized into components and the interfaces between those components”. Similarity between the two definitions is clear if we consider that interactions among agent roles are realized in form of communications and their specification constitutes the agent interfaces.

The holonic paradigm corresponds to the adoption of the holon as the founding element of the architectural style. In the adoption of the holonic architectural style we choose to organize the MAS according to “a self-similar or fractal structure that is stable, coherent and that consists of several holons as sub-structures and is itself a part of a greater whole” [15].

This lead us to redefine the kinds of coupling as it is shown in Table 1 column 3. However these definitions are still too general for allowing us to define metrics for evaluating HMAS coupling. It might seem that the concept of holon may be directly mapped to the one of module (such as for AO context) but it not so because of some intrinsic aspects of a holon, such as its duality that we will detail later in the paper.

In the following subsection we provide some details about holonic multi-agent systems in order to detail and clear the mapping between objects and holonic agents concepts.

### 3.2 Redefining Concepts from the HMAS Perspective

The concept of Holon is based on the so called Janus ef-

<table>
<thead>
<tr>
<th>Object Oriented Paradigm</th>
<th>Agent Oriented Paradigm (Magarino et.al)</th>
<th>Holonic Paradigm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Description</td>
<td>Name</td>
</tr>
<tr>
<td>Data Type Coupling</td>
<td>Two modules use the same data type</td>
<td>Ontology Sharing</td>
</tr>
<tr>
<td>Data Coupling</td>
<td>Data from one module is used in another</td>
<td>Knowledge coupling</td>
</tr>
<tr>
<td>Control Coupling</td>
<td>One module may control actions of another</td>
<td>Behavioural coupling</td>
</tr>
<tr>
<td>Content Coupling</td>
<td>A module refers to internals of another module</td>
<td>Inner structural coupling</td>
</tr>
</tbody>
</table>

**Table 1: Kinds of Coupling in OO, AO and Holonic Architectures**
that is a single instance of a moderated group representing the government of the super-holon. It specifies how members govern and manage the super-holon. All members of the super-Holon must belong to this group. (ii) At least a Production Group relates to the problem the members are collaborating to solve. This group may contain only a subset of the members of the super-holon. It describes how members interact and coordinate their actions to fulfill the super-holon tasks and objectives. It is important to underline that at the finest level of granularity, a holon is atomic, and in HMAS it may be considered as a classical Agent. As a consequence, it does not contain any group.

All these HMAS concepts have to be mapped onto the subsystems/modules and components concepts; Parnas definition and the Janus effect guided us in doing this. The Janus effect can be observed in all the aspects of a holon. Indeed, if we consider the whole holarchy a holon is a whole from the viewpoint of the holons at the same holarchy level and a part from the viewpoint of its super-holon. If we consider the organizational schema, a holon can be seen from the government perspective, looking at its holonic group, and from the production perspective, looking at its production groups. Finally, both in the two aspects, government and production, a holon can be seen from the social structure perspective, looking at the roles played by the member of the production group and their interactions, and from the functional perspective, looking at the goals that the roles of the production group have to satisfy.

The metrics for HMAS we propose take into account these two overlapping aspects. Hence, we define metrics for the government and production aspects from both the social and functional perspective. As a consequence, the concepts for module, component, internal and external dependencies we have identified in HMAS are related to these features. In particular, from the social perspective of government a module is the Holonic Group whilst a component is a Role. Roles depend each other by means of communications. From the government functional perspective, a module is also in this case corresponding to the Holonic Group, whilst a component is a Goal. Goals may depend on another goal for their satisfaction. For space concerns, in Table 2 we list all the concepts useful for defining HMAS metrics.

4. GOVERNMENT METRICS

There is a wide range of government configurations for a super-holon starting from Apanarchy to Monarchy [11][9]. For the scope of this paper, we consider the most common forms of governance adopted in HMASs (see Table 3).

<table>
<thead>
<tr>
<th>Government Type</th>
<th>Social Structure</th>
<th>Responsibilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monarchy</td>
<td>1 Head, 0 Peer</td>
<td>Exclusive command of the Head</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Peers depend on Head’s decisions</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No social exchange/activities between Head and Peer</td>
</tr>
<tr>
<td>Oligarchy</td>
<td>N_H Head, N_Peer</td>
<td>Command shared by a group of Heads</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Peers depend on Heads’ decisions</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No social exchange/activities between Head and Peer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>All Heads are engaged in social exchange/activities</td>
</tr>
<tr>
<td>Polyarchy</td>
<td>N_H Head, N_Peer</td>
<td>Command shared by a groups of Heads</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Peers depend on Heads’ decisions</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Heads depend on Peers for certain decisions</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Peers and Heads are involved in social exchange/activities</td>
</tr>
<tr>
<td></td>
<td></td>
<td>All Heads are engaged in social exchange/activities</td>
</tr>
<tr>
<td>Apanarchy</td>
<td>0_H Head, 0 Peer</td>
<td>Command is shared by all members</td>
</tr>
<tr>
<td></td>
<td></td>
<td>All Heads are engaged in social exchange/activities</td>
</tr>
</tbody>
</table>

Table 3: Government Forms.
Table 4: Government Metrics

<table>
<thead>
<tr>
<th>Government Type</th>
<th>N_{a=5}</th>
<th>Social Cohesion</th>
<th>N_{a=10}</th>
<th>Social Cohesion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monarchy</td>
<td>1 Head, 4 Peer</td>
<td>0</td>
<td>1 Head, 9 Peer</td>
<td>0</td>
</tr>
<tr>
<td>Oligarchy</td>
<td>2 Head, 3 Peer</td>
<td>0.1</td>
<td>2 Head, 8 Peer</td>
<td>0.02</td>
</tr>
<tr>
<td>Polyarchy</td>
<td>2 Head, 3 Peer</td>
<td>0.3</td>
<td>3 Head, 7 Peer</td>
<td>0.06</td>
</tr>
<tr>
<td>Apanarchy</td>
<td>5 Head, 0 Peer</td>
<td>1</td>
<td>10 Head, 0 Peer</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 5: Social Cohesion for government type of two different super-holons.

These results show a set of cohesion values coherent with the kind of governance. Indeed, in the range of the government types, the social engagement grows according to the decentralization of the command, which increases the social exchanges. This means that a monarchic holonic group reduces the modularity of its super-holon conversely from the apanarchy. Moreover, as expected the cohesion of oligarchic and polyarchic government with 10 members is less than the one with 5 members. This is because there are more components (i.e. Peers) not involved in the decision process.

Functional Perspective.

The functional perspective for a holonic group concerns the goals its roles have to satisfy in order to govern its super-holon. For the sake of clarity, we use the term goal for indicating goals of Roles, social goals for goals of the holonic group and super-holon goal for indicating the goal of the super-holon the holonic group belongs to. We assume that each Role may contribute with its own goals to the achievement of the social goals.

Let us suppose to have a composed holon (super-holon $SH$) with $n_M = n_{Peer} + n_{Head}$ members. Analogously to social perspective, we have to determine the external and the internal dependencies in order to define functional coupling and functional cohesion for the $SH$’s holonic group (see eq.1 and eq.2). By definition (see Table 2), the holonic group is decoupled from the super-holon goal when different types of governance are adopted.

$FCh_{Monarchy} = 0$ (4)

$FCh_{Oligarchy} = \frac{k}{k + (k - 1)n_{Peer}}$ (6)

$FCh_{Polyarchy} = \frac{k + w * n_{Peer}}{k + (k - 1)n_{Peer}}$ (8)

$FCh_{Apanarchy} = 1$ (10)

Social Cohesion

Let us consider a composed holon (super-holon, $SH$) with $n_M = n_{Peer} + n_{Head}$ members. In order to define coupling and cohesion for the $SH$’s holonic group we have to determine the external and internal dependencies (i.e: communications) among members. Looking at Table 3 we can see that: (i) in the Monarchy there are no social exchanges because all Peers depend on the decisions of the single Head; (ii) in an Oligarchy all Heads (and only them) are engaged in social exchanges (hence: $IntDep = n_{Head}*(n_{Head}−1)$); (iii) in a Polyarchy, social exchanges are established both among Heads and between Peers and Heads (i.e: $IntDep = n_{Head}*(n_{Head}−1) + n_{Peer} * n_{Head}$); (iv) finally the internal dependencies for Apanarchy are $IntDep = n_{Head}*(n_{Head}−1)$ because all members are Heads involved in mutual social exchanges. Hence, according to eq.1 and table 2, the social cohesion for each government configuration is given by the equations 3, 5, 7, 9 listed in Table 4.

In order to give an example, we assume to have a super-holon with five members and another with ten members. According to equations 3, 5, 7 and 9, Table 5 shows the cohesion values of the holonic groups of these super-holons when different types of governance are adopted.

Table 5: Social Cohesion for government type of two different super-holons.

1Hereafter, the prefix S or F differentiates social metrics from functional ones.
is given by eq.11 in Table 7. Moreover, let us suppose that $N_G$ is the number of production groups inside a super-holon, the social cohesion of the production as a whole is given by eq.13 in Table 7.

From the social perspective, conversely the coupling measures the dependencies among production groups of a super-holon. Let $N_R$ and $N_G$ be the number of roles and goals inside the $i$-th and $j$-th production group, their coupling is given by eq.12 in Table 7. Whilst, let us suppose that $N_G$ is the number of production groups inside a super-holon, the social coupling of the production as a whole is given by the eq.14 in Table 7 where $\binom{N_G}{2}$ is the number of possible pairs of groups.

**Functional Perspective.**

The functional perspective for a production group concerns the goals its roles have to satisfy in order to reach the social goals. The social goals of a production group are goals that contribute to the achievement of super-holon goal. As previously said, we use the term goal for addressing the goal of a Role, the term social goal refers to goals of a production group and super-holon goals to goals of the super-holon the production groups belong to.

Let us suppose $N_SG$ be the number of social goals related to a production group $PG$, $N_R$ the total number of goals of Roles and $GoalDependency_{ij}$ the internal dependencies between the $i$-th social goal and $j$-th goal, then the $PG$ functional cohesion is given by the eq.15 in Table 7. Whilst, let $NG$ be the number of production groups inside a super-holon, the functional cohesion of the production as a whole is given by the eq.17 in Table 7. Analogously, we define the functional coupling between the $i$-th and the $j$-th production groups as reported in eq.16 in Table 7, where $N_SG$ and $N_SG_i$ are the number of social goals related to the $i$-th and $j$-th production group of the super-holon and $GoalDependency_{ij}$ is the external dependency between $k$-th social goals of the $i$-th production group and the $w$-th social goal of the $j$-th production group.

Whilst, let $NG$ be the number of production groups inside a super-holon, the functional coupling of the production as a whole is given by eq.18 in Table 7 where $\binom{NG}{2}$ is the number of possible pairs of groups.

**Applying the proposed metrics in a design problem.**

Let us suppose to design only the production part of a super-holon for solving the following problem: to unload goods from a lorry in order to send them toward new destinations. We can do this by adopting different design solutions for the super-holon, for our purposes we might consider three different organizational structures (see Fig.1).

![Figure 1: Social design choices for production groups.](image-url)
In the first case (see Fig.1 a)), we designed one super-holon with only one production group including four roles: an AGV role devoted to unload goods from a lorry and to transport them to the sorter; a Route Planner that manages the routes the AGV uses to move toward/from the lorry to be unloaded; a Sorter that packages goods and gives them to the proper Truck (see next role), and finally a Truck that transports packages to their destination (several trucks leave the yard at the same time, the Sorter sends each parcel to the right one). Moreover, we established that four communication links are necessary among these roles for exchanging information. In the second case, we designed the super-holon with two production groups (see Fig.1 b)) including five roles. In so doing, we have extracted the packaging tasks from the previously defined Sorter role and we have delegated them to a new role, the Packager. As a result, the production group C is responsible for unloading cargo and packaging accordingly to destination while production group B is responsible for delivering packages to their destination. Finally in the third case, we designed the super-holon with three production groups (see Fig.1 c)) with six roles, where we have extracted the task of unloading goods from the AGV role delegating them to a new role labelled Unloader; the AGV role remains responsible for delivering goods to the Packager only. Fig.2 shows three different functional design choices related to the three previous organizational structures. For space concerns, we omit the description of each goal also because for our purposes we are interested only to the number of goals and their dependencies.

The resulting values of social/functional cohesion and coupling are shown in Table 8. It is worth to note that these values are not very useful when interpreted as absolute values on a single design solution. They become meaningful for making a choice among different alternatives. Specifically, in accord to quality requirements prescribed by software specifications of the system to be realized, the designer can make comparisons among different architectural styles by using these values as guidelines to choose appropriate design solutions. For example, the design solution represented by the case c) of Fig.1 may provide a better modularity than the others, because it has the greatest social and functional cohesion and the same social and functional coupling of the case b) (see Table 8). Vice versa, although the design solution represented by the case a) has a zero coupling it is not a good choice for modularity, because everything is encapsulated within a single great module. Naturally, the choice of case c) is not always the right solution to be adopted. It depends on the requirements of the system to be realized.

### Table 7: Production Metrics

<table>
<thead>
<tr>
<th>METRICS</th>
<th>Case a)</th>
<th>Case b)</th>
<th>Case c)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SChProdGroup</td>
<td>0.083</td>
<td>0.083</td>
<td>0.083</td>
</tr>
<tr>
<td>FChProdGroup</td>
<td>0.083</td>
<td>0.125</td>
<td>0.125</td>
</tr>
<tr>
<td>SCProd</td>
<td>0</td>
<td>0.125</td>
<td>0.125</td>
</tr>
<tr>
<td>FCPProd</td>
<td>0.42</td>
<td>0.75</td>
<td>0.66</td>
</tr>
<tr>
<td>SChProduction</td>
<td>0</td>
<td>0.083</td>
<td>0.083</td>
</tr>
<tr>
<td>FChProduction</td>
<td>0.42</td>
<td>0.875</td>
<td>0.89</td>
</tr>
<tr>
<td>SCProduction</td>
<td>0</td>
<td>0.125</td>
<td>0.125</td>
</tr>
<tr>
<td>FCPProduction</td>
<td>0</td>
<td>0.75</td>
<td>0.75</td>
</tr>
<tr>
<td>Social Coupling</td>
<td>SCProd = ∑ ∑ GoalDependencyij / NjG * NiG</td>
<td>SCProd = ∑ ∑ GoalDependencyij / NjG * NiG</td>
<td>SCProd = ∑ ∑ GoalDependencyij / NjG * NiG</td>
</tr>
<tr>
<td>Functional Coupling</td>
<td>FCPProd = ∑ ∑ GoalDependencyij / NjG * NiG</td>
<td>FCPProd = ∑ ∑ GoalDependencyij / NjG * NiG</td>
<td>FCPProd = ∑ ∑ GoalDependencyij / NjG * NiG</td>
</tr>
</tbody>
</table>
applied in order to determine the cohesion and coupling for a given (super-)holon. Now, in order to give a complete view on the entire holonic architecture, it is necessary to further define the metrics for determining the cohesion and coupling measures for the whole holarchy. We deduced those directly from the previous ones.

Let us consider a holarchy with $N_{SH}$ super-holons, the social cohesion of the whole holonic architecture is given by:

\[
S_{Ch_{a}} = \sum_{w=1}^{N_{SH}} \frac{S_{Ch_{w}}}{N_{SH}}
\]

(19)

\[
S_{Ch_{SH}} = \frac{\sum_{k=1}^{N_{G}} S_{Ch_{k}} + S_{Ch_{HolonicGroup}}}{N_{G} + 1}
\]

(20)

where $S_{Ch_{SH}}$ is the social cohesion of a super-holon, $N_{G}$ is the number of production groups inside the super-holon, $S_{Ch_{k}}$ is the cohesion for each production group given by eq.11 and $S_{Ch_{HolonicGroup}}$ is the cohesion for the holonic group of the super-holon given by one of eq.3 - 9 according to the type of government adopted for the super-holon.

Analogously, let us consider a holarchy with $N_{SH}$ super-holons, the social coupling of the whole architecture is given by:

\[
S_{Cp_{a}} = \sum_{w=1}^{N_{SH}} \frac{S_{Cp_{w}}}{N_{SH}}
\]

(21)

\[
S_{Cp_{SH}} = S_{Cp_{Production}}
\]

(22)

where $C_{p_{SH}}$ is the social coupling of a super-holon equals to $S_{Cp_{Production}}$ because $C_{p_{HolonicGroup}} = 0$. As concerns functional cohesion and coupling of a holarchy, they are analogous to the previous ones.

It is also worth to point out that these metrics can be effectively employed as a part of a methodological approach for designing holonic systems. For instance, by using an iterative and incremental approach these metrics may provide the designer a step by step measure of the satisfaction of quality requirements imposed for the system to be realized. Moreover, assuming the metrics integrated in a CASE tool for designing holonic systems, a designer could be able to immediately know the impact of an architectural change on the whole system thus having the possibility to make a correction if the new choice should negatively affects some quality requirements. By now, we are working on a suite of metrics for evaluating other quality attributes for HMASs in order to provide useful means for supporting design choices. This metrics suite will be incorporated into a HMAS design tool that will provide functionalities for automatically calculating them.

**Acknowledgments**

This work has been partially founded by PO-FESR Sicilia 2007-2013 Project OCCP-Open Cloud Computing Platform a misura di organizzazioni agili.
REFERENCES


