Software Agents for Coalition Forces

Zakaria Maamar
Software Agents Research Group$_{@ZU}$
College of Information Systems
Zayed University
PO Box 19282, Dubai, U.A.E
zakaria.maamar@zu.ac.ae

Paul Labbé
System of Systems Section
Defence Research Establishment Valcartier
2459 Pie-XI Blvd. North Val-Bélair
(Québec), Canada, G3J 1X5
paul.labbe@drev.dnd.ca

Wathiq Mansoor
Software Agents Research Group$_{@ZU}$
College of Information Systems
Zayed University
PO Box 19282, Dubai, U.A.E
wathiq.mansoor@zu.ac.ae

Abstract. The distributed, heterogeneity, and dynamic nature of the coalition context has raised the need for new advanced technologies. These technologies aim at managing the coalition informational infrastructure, in terms of autonomy, adaptability, and scalability. To achieve this support, Software Agents (SAs) seem to be a promising approach. To develop this approach, different aspects of a coalition has to be identified. These aspects include the coalition structure; the roles and responsibilities held by people within the coalition; the flow of information within the coalition; the capabilities required or available within the coalition; and the context in which the coalition operates. For many of these aspects, SAs can be used; . For instance, the coalition structure can be associated with several SAs of different types and with different roles.

1. Introduction

We discuss our research work on the design and development of collaborative environments for distributed and heterogeneous military applications. These applications, called Command & Control Information Systems (CCISs), are increasingly important for land, naval, and air operations. Moreover, CCISs have civilian applications in multiple areas such as air traffic control, search & rescue, and emergency services. In a military context, a commander makes decisions concerning his troops deployment using the information supplied by the CCIS. It may occur that this commander aims at involving other friendly CCISs before taking his decisions. For example, a Canadian commander has to take into account the positions of the enemy and friendly troops. Therefore, he has to involve other CCISs that may possess such an information. It would be more appropriate if this commander could perform this operation without being aware of each CCIS's characteristics. Take for instance a situation where different countries decide to set up a coalition for an international humanitarian assistance. In fact, the CCIS of each country has its own functional and structural characteristics. It is impossible for a commander to be aware of all the CCISs' locations, languages, and information semantics. Therefore, it becomes urgent to propose new support technologies that will free military users from worrying about the distributed, heterogeneous, and dynamic nature of the coalition, in general and CCISs in particular. In this paper, we describe the IC2MAS (Interoperable Command & Control based on MultiAgent Systems) project that aims at managing the coalition infrastructure at the following levels (adapted from [Babin et al., 1994]):

- Autonomy: in the coalition environment, CCISs should have the flexibility to be designed, developed, and managed independently, without having to comply with this environment's standards.
- Flexibility: CCISs, that use either standard or non-standard technologies, as well as new and legacy CCISs, should be incorporated into the coalition environment in a "seamless" way without causing any disruption to this environment.

- Scalability: the total coalition environment should be expandable by allowing this coalition to start with a number of countries and gradually extend over time, without loosing integrity.

Taking into account these three levels and the requirements of a coalition (discussed in Section 3.1), the IC2MAS project has established an interoperability approach to provide effective support to a coalition.

The motivation behind the support of a coalition is to provide an integrated view of all the aspects that are relevant to this coalition. These aspects are multiple and include: the coalition structure; the roles played by people and responsibilities held by them within the coalition; the flow of information within the coalition and with the external world; the resources required by and available within the coalition; and the context (war, peace-making/keeping, from war to peace-making, and from peace-keeping to war) in which the coalition takes place. MASs could handle a number of these aspects. For instance, the coalition structure could be viewed as a collection of collaborative MASs; each MAS could correspond to a CCIS and each MAS could contain different types of agents, fulfilling different roles, and carrying out different responsibilities.

In the IC2MAS interoperability approach, MASs are the CCIS's front-ends to the coalition network and hence, have the capability to act on their behalf. Moreover, MASs encompass different Software Agents (SAs) [Green et al., 1997] that handle and perform the functionalities required to coalition support, for example managing the CCISs' autonomy and invoking CCISs. However, given the distributed nature of a coalition and the network features in terms of reliability and bandwidth capacity (e.g. the coalition could occur in a country in which the network infrastructure is not well developed), some of the SAs in the IC2MAS approach are able to create Slave-Agents [Buschmann et al., 1996] and enhance them with mobility mechanisms [Lange and Oshima, 1999]. A mobile agent can move from one system to another to perform specific operations, instead of continuously keeping the network "busy". Moreover, it often happens that SAs have to work together to execute common operations. For instance, in a coalition, the Canadian forces have to interact with non-government organizations as well as with armed forces of other countries. Therefore, SAs have to rely on communication [Labrou et al. 1999a] and coordination [Hamada, 1997] mechanisms to avoid conflicts and collaborate efficiently. When diverse SAs communicate, they have to understand each other. By establishing an ontology [Jones et al., 1998], a common terminology and semantic basis for the various SAs is offered. Hence, the risk of getting inconsistent information is reduced.

The paper is organized as follows. Section *1* proposes an overview of our theoretical, i.e. MASs, and practical, i.e. CCISs coalition, research project. Section *2* presents the degrees of interoperability in a coalition and an overview of the CCISs field. Section *3* describes the characteristics of the IC2MAS interoperability approach. Section *4*, briefly, reviews the related work. Section *5* gives insights on topics that are currently, tackled. Finally, Section *6* consists of concluding remarks.

2. Background

This section is divided into two parts. The first part identifies the degrees of interoperability in a coalition while the second part provides an overview of CCISs.

2.1 Interoperability Degrees in a Coalition

In a coalition, three degrees of interoperability are identified (adapted from [Au et al., 1999]). Basic interoperability, called interconnectivity, allows simple data transfer (with no semantic), whereas application-level integration enables applications (for example, CCISs) running in any environment to exchange services and perform computing, even if these applications were designed at different times by different persons. In a coalition, working at the application-level is not enough, particularly if the military forces aim at merging their operational processes. Therefore, a collaboration at the commandment level is required. In what follows, the three degrees of interoperability are summarized (cf. Figure 1):

- Physical interconnectivity: to guarantee basic communication, computing resources are first interconnected to exchange messages. This interconnectivity occurs at the physical level.
- Application integration: its main purpose is to carry out operations among different computing resources. Generally, these resources are distributed across networks and heterogeneous at different levels (hardware, software, and terminology).
- Commandment collaboration: it goes beyond application integration, by expanding military

operational processes to other structures. To this end, a collection of components, such as software agents gathered into multiagent systems, could be set up. These components collaborate more than just interoperate.

In Figure 1, the commandment level relies on the application level to achieve the coalition mission, for example humanitarian assistance. Furthermore, the commandment level interacts regularly with its headquarter. The purpose is to keep the headquarter informed about the progress of the mission. In order to assist the commandment level in its daily operations, the application level offers different types of services, such as data fusion and logistic. In fact, the application level is built on top of the physical level and hence, uses its computing resources. When the coalition's military forces have to collaborate, they go through a coordination process. Such a process could be entrusted to their respective MASs. In order to collaborate efficiently, military forces have to agree on how to invoke mutually their services. To this end, their respective applications have to be integrated.

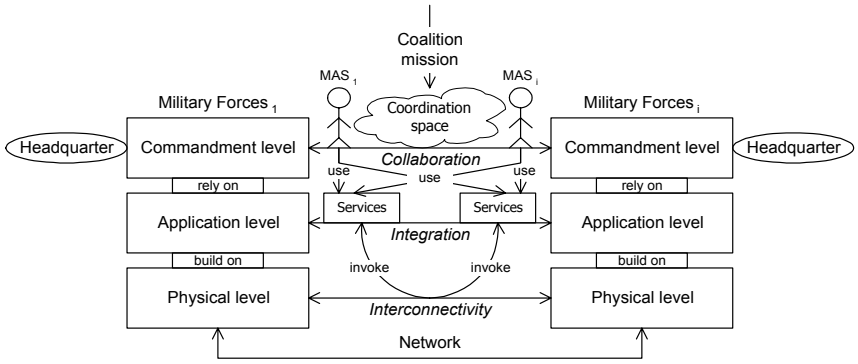

Figure 1 From interconnectivity to collaboration, through integration

2.2 An overview of CCISs

Nowadays, information technologies are an inherent part of the commanders' decision-making process. Particularly, CCISs help commanders to obtain a view of the tactical situation in which they are involved. In fact, a CCIS is used to gather information from different sensors, process this information, and suggest actions to be taken by the commander. Hence, CCISs are crucial and should meet demanding criteria in terms of reliability, efficiency, and fault-tolerance.

According to [Malerud et al., xxxx], a CCIS consists of a structure, functions, and tasks. The CCIS structure represents an assembly of facilities, arranged to meet the CCIS's objectives. To reach these objectives, the CCIS's functions are initiated in order to carry out the needed tasks. Tasks require the structure's facilities, in terms of personal, technical equipment, computing time, and so on. Figure 2 presents a simplified architecture of a CCIS. Several types of functions exist within the CCIS, ranging from planning and weather forecast to data fusion. These functions are offered to users and are built on top of a support structure in terms of hardware and software resources. Furthermore, some of these functions receive messages from the external environment, e.g. remote sensors, through a communication module. Currently, multiple definition languages of messages, e.g. USMTF, are available. These languages allow formatting messages in order to be automatically parsed by appropriate engines of the different functions. Unfortunately, such languages cannot be used in the achievement of interoperable CCISs. These languages' structures are too rigid and do not have semantics.

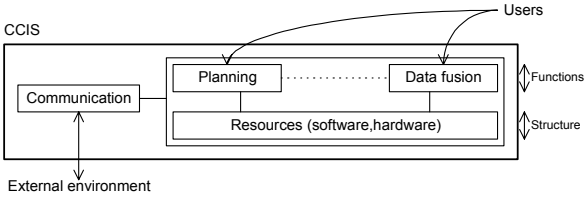

Figure 2 CCIS Simplified Architecture

As CCISs are getting larger and more complex, their interoperability and hence collaboration, in a coalition

context for example, are becoming a central concern for military users and CCISs' designers. Therefore, the IC2MAS project aims at handling this concern, by providing 1) users with services that will free them from worrying about the characteristics of the interconnected CCISs; and 2) designers with approaches based on advanced technologies, such as MASs.

3. Presentation of the IC2MAS Approach

This section presents the IC2MAS approach. First, major requirements of a coalition are described. Next, IC2MAS architecture and types of SAs are presented. Finally, IC2MAS operating is detailed.

3.1 Coalition's Requirements

In the IC2MAS project, the running scenario corresponds to a coalition that is set up by different countries for different purposes: international humanitarian-assistance, peace support operations, etc. The coalition scenario is appropriate for several reasons:

- People from different countries, at different locations, and at different moments contribute to the definition of the same operations, for instance deploying troops in a combat zone. However, these people do not use the same communication language and do not manage the same types of resources that vary from high to low technologies. It happens that certain countries are well equipped than others.
- At diverse hierarchical levels, different people take decisions during the performance of operations. It happens that a decision is based on an information that is not well understood by all people. Moreover, it happens that a decision requires the interaction of diverse CCISs that could be distributed and heterogeneous.
- At the theater of operations, it is complex to provide and maintain a high level of assistance to military users. For example, it is not possible to afford to each combat unit an expert in PC software, an expert in Unix software, etc. Moreover, it is not possible for a military user to be aware of the characteristics of the different CCISs of the coalition.

Major requirements to coalition support constitute a framework that identifies what types of information could be exchanged, what types of operations could be delegated, and what types of communication approaches could be used. In what follows, a research avenue is associated with each requirement.

- Requirement: What types of information could be exchanged?
  Research Avenue: Ontology.
  Definition: an ontology is a means to express and exchange information that is understood by all the participants of the coalition. Moreover, to be used efficiently an ontology requires a language to be represented, e.g. KIF, and a language to be communicated, e.g. KQML.
- Requirement: What types of operations could be delegated?
  Research Avenue: SAs integrated into MASs.
  Definition: a SA is an autonomous, goal-oriented entity that has the ability to assist users in performing their tasks, to collaborate with other agents (software or human) to jointly solve problems, and to answer users' needs. Furthermore, a collection of SAs can be gathered into a MAS architecture. As stated in [Labrou et al. 1999ba], communities of agents are much powerful than any individual agent.
- Requirement: What communication approaches could be used?
  Research Avenue: Remote/Local communication.
  Definition: Communication between distributed components, for example SAs, could occur either remotely or locally. In the latter case, the components have to move to a common workplace.

3.2 IC2MAS Architecture

In the literature, different approaches that deal with the problem of interoperable systems can be found, among them Infosleuth [Bayardo et al., 1997], TSIMMIS [Chawathe et al. 199], SIMS [Knoblock et al., 1997], and SIGAL [Maamar et al., 1999]. All these approaches agree on the use of SAs, as a means to develop such systems and have several elements in common, such as all the SAs are static. Therefore, these SAs do not have the opportunity to move to distant systems. Moreover, all these approaches assume that the network infrastructure is fully reliable and has unlimited bandwidth for information transmission.

Based on these different approaches and the coalition's requirements, we proposed an IC2MAS architecture to the coalition (cf. Figure 3). Multiple MASs form the backbone of this architecture and interact remotely as well as locally. In addition, these MASs collaborate through a facility called Advertisement Infrastructure. It is managed by an agent and contains a Bulletin Board and a Repository of Active-Agents. Currently, we are aware that the Advertisement Infrastructure could be considered as a bottleneck. However and in the mid-term, this infrastructure could be duplicated and spread across networks.

In the IC2MAS architecture, MASs integrate different types of SAs: Interface-Agents assisting users, CCIS-Agents invoking CCISs' functions and satisfying users' needs, Resolution-Agents, also, satisfying users' needs, Control-Agents managing MASs, and finally, a Supervisor-Agent managing the Advertisement Infrastructure. In the IC2MAS environment, the Resolution-Agent is able to create Slave-Agents and transmit them either to the Advertisement Infrastructure or to other distant MASs. Slave-Agents carry out operations on behalf of Resolution-Agents. Slave-Agents' creation process complies with the Supervisor-Worker pattern as defined in [Fischmeister and Lugmayr, 1999]. In the next sections, agents' functionalities are depicted.

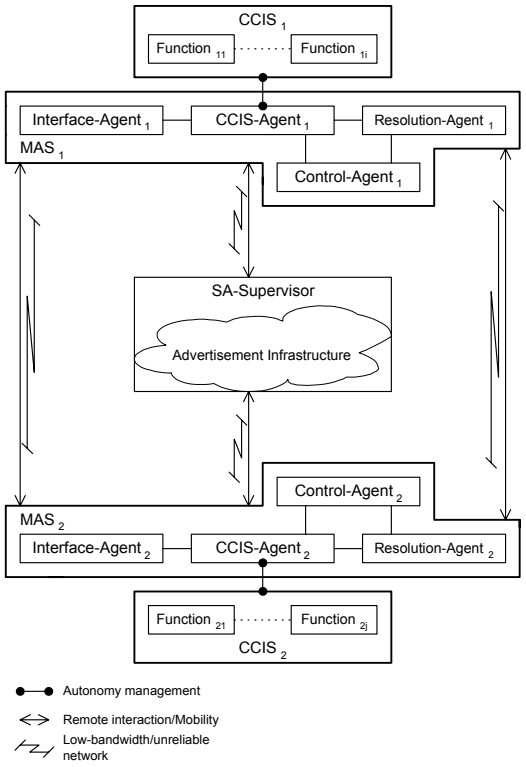

Figure 3 IC2MAS Architecture for coalition support

3.3 Software Agents and Advertisement Infrastructure

Different types of SAs exist in the IC2MAS architecture. These SAs belong to different MASs and collaborate through the Advertisement-Infrastructure facility. In what follows, certain agents' internal-modules are detailed.

Interface-Agent - By analogy to Interface-Agents of [Maamar et al., 1999, Sycara et al., 1996], the IC2MAS's Interface-Agent assists users in formulating their needs, maps these needs into requests, forwards these requests to the CCIS-Agent in order to be processed, and provides users with answers obtained from the CCIS-Agent.

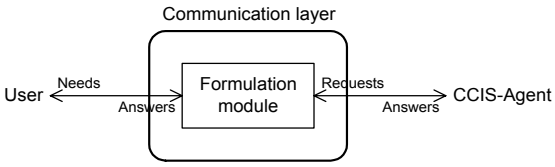

Figure 4 Interface-Agent modules

The Interface-Agent consists of one module, called formulation that is encapsulated into a communication layer (cf. Figure 4). The formulation module takes as inputs users' needs and CCIS-Agent's answers and provides as outputs requests to CCIS-Agents and answers to users. In the IC2MAS environment, users describe their needs according to the concepts that are understood by Interface-Agents (cf. Section 5.1).

CCIS-Agent - By analogy to Resolution-Agents of [Maamar et al., 1999] and Task-Agents of [Sycara et al., 1996], the IC2MAS's CCIS-Agent processes users' requests, only if these requests need the involvement of the CCIS of this particular CCIS-Agent. These requests are transmitted by the Interface-Agent. In addition, and by analogy to Knowledge-Agents of [Maamar et al., 1999], the IC2MAS's CCIS-Agent acts on CCIS behalf and hence, maintains its autonomy towards the coalition. To achieve this autonomy, the CCIS-Agent advertises, through its services (currently, the services do not have constraints, e.g. cost), the functions its CCIS performs. Here, the term service denotes a computing procedure, for example requesting the CCIS's weather-forecast function. In the IC2MAS environment, a CCIS-Agent advertises its services, by posting notes on the Bulletin Board of the Advertisement Infrastructure. In fact, the CCIS-Agent sends remote requests to the Supervisor-Agent. Before posting notes, the Supervisor-Agent checks the CCIS-Agent's security level to authenticate this CCIS-Agent's requests and identify the services it is authorized to advertise.

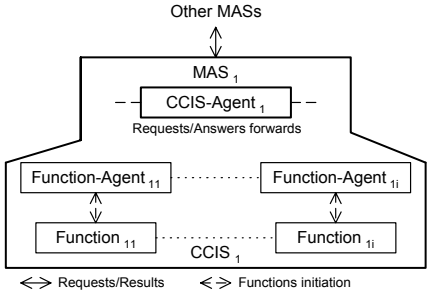

Figure 5 Function-Agents at the MAS level

A CCIS offers different functions that vary from data fusion and weather forecast to planning (cf. Section 2.2). Based on these functions and the complex nature of CCISs, for instance a planning function could be a distributed-object client/server application running on top of an Object Request Broker middleware, new types of SAs, called Function-Agents, are introduced in the IC2MAS architecture, and particularly at the MAS level. Each Function-Agent is associated with a CCIS's function. As a result, a CCIS-Agent manages a group of Function-Agents that evolves under its supervision (cf. Figure 5). For instance, a request to the planning function of a CCIS is initially, sent to the CCIS-Agent that forwards this request to the appropriate Function-Agent. Hence, a Function-Agent knows the protocols through which a function of a CCIS accepts requests and provides back results. IC2MAS's Function-Agents are similar to Information-Agents of [Sycara et al., 1996].

Figure 6 presents CCIS-Agents' and Function-Agents' modules. As the Interface-Agent, a communication layer encapsulates both agents' modules. The CCIS-Agent consists of two modules: definition and pre-processing. The IC2MAS administrator uses the definition module. He specifies the services to be advertised by the CCIS-Agent. The pre-processing module identifies whether or not the CCIS of a CCIS-Agent could satisfy users' requests. If not, these requests are transmitted to the Resolution-Agent. The pre-processing module relies on an information source, called CCIS capabilities. Moreover, the administrator updates this information source with the services it has specified. The Function-Agent consists of two modules: processing and monitoring. The processing module receives requests from the CCIS-Agent and

performs them against the CCIS's function. The monitoring module monitors the modifications that could occur at the CCIS's functions level. These modifications have to be notified to the CCIS-Agent's definition-module.

Resolution-Agent - By analogy to Resolution-Agents of [Maamar et al., 1999] and Task-Agents of [Sycara et al., 1996], the IC2MAS's Resolution-Agent processes users' requests, only if these requests are transmitted by the CCIS-Agent and need the involvement of several CCISs to be completed. In fact, the resolution process requires that the Resolution-Agent collaborates with the CCIS-Agents of other MASs, including or not the CCIS-Agent of this Resolution-Agent's MAS.

At IC2MAS start-up time, the Resolution-Agent creates a Slave-Agent, called Help-Agent, and sends it to the Advertisement Infrastructure. As soon as the Help-Agent arrives, the Supervisor-Agent checks it. Next, the Help-Agent waits for the Resolution-Agent's queries about the services to look for the Bulletin Board[1].

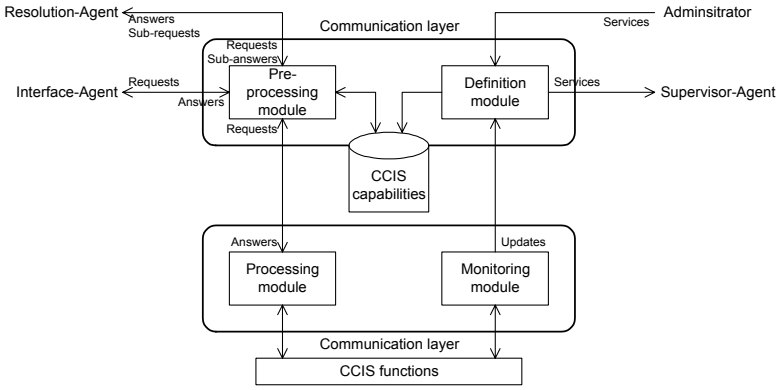

Figure 6 CCIS-Agent and Function-Agent modules

In order to identify the CCIS-Agents that are required to satisfy users' requests, the Resolution-Agent sends remote queries to the Help-Agent. This agent browses the Bulletin Board, identifies appropriate CCIS-Agents through their offered services, and finally informs remotely its Resolution-Agent parent. Next, the Resolution-Agent designs the procedure needed to the performance of the user's request. Generally, such a procedure is called a route or an itinerary. Then, the Resolution-Agent creates another Slave-Agent, called Route-Agent, and assigns to it this procedure. The Route-Agent may require either interacting remotely with the CCIS-Agents of the other MASs or migrating to the MASs and meet locally their CCIS-Agents. A decision about a remote request or mobility is based on the network status and the number of the CCISs required satisfying users' requests[2]. As CCIS-Agents, a security level is also associated with Slave-Agents. This security level is used to check Slave-Agents entering the Advertisement-Infrastructure as well as the different MASs.

The Resolution-Agent consists of two modules, called slave and pre-processing (cf. Figure 7). Both of them are encapsulated into a communication layer. The slave module creates Slave-Agents, namely Help-Agent and Route-Agent. The pre-processing module designs the procedure that is used to perform users' requests. This procedure is forwarded to the Route-Agent's performance module. This agent carries out these requests, according to the CCISs that have been identified by the Help-Agent's browsing module.

1 A Help-Agent could regularly consult the Bulletin Board in order to inform its Resolution-Agent about the notes that could interest it.

2 It is stated in [Bredin et al., 1999] that the value of mobile-agent system depends on both the number of host sites that an agent might migrate to as well as the number of other agents with which an agent may interact.

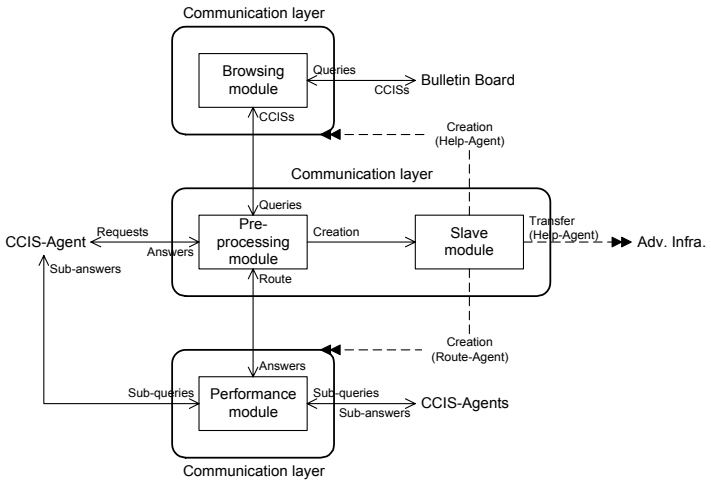

Figure 7 Resolution-Agent modules (including Help-Agent and Route-Agent)

Control-Agent - In an environment consisting of mobile agents, mobility operations consist of shipping the agents through the net to other distant systems, authenticating these agents as soon as they arrive, and finally installing these agents to resume their operations. In the IC2MAS environment, the Control-Agent of the MAS is in charge of all these operations. For instance, when a Help-Agent moves, it first interacts with the Control-Agent in order to be shipped to the desired MAS. Furthermore, Control-Agents maintain the coherence of their MASs by keeping track of the Route-Agents entering and leaving these MASs.

Supervisor-Agent - A Supervisor-Agent is in charge of several operations. It manages the Advertisement Infrastructure by receiving CCIS-Agents' advertisements, sets up a security policy in order to monitor the Help-Agents accessing this infrastructure, and finally, installs Help-Agents to resume their operations in this infrastructure.

In the IC2MAS environment, the Supervisor-Agent uses the Repository of Active-Agents to register all the Help-Agents and CCIS-Agents that have respectively got an agreement to enter the Advertisement Infrastructure and advertise their services. The Repository of Active-Agents is, also, updated when Resolution-Agents decide to remove their Help-Agents from the Advertisement Infrastructure.

Advertisement Infrastructure - In a coalition context, CCISs are spread across networks and generally rely on low-bandwidth and/or unreliable channels for communications. Moreover, a military user may use his VHF Combat Net Radio to send and request information. This military usually relies on mobile devices, such as portable computers, that are only intermittently connected to networks. In the IC2MAS environment, instead of overloading the network, Help-Agents migrate to the Advertisement Infrastructure and browse locally the Bulletin Board, looking for appropriate CCISs.

3.4 IC2MAS operating

Based on the characteristics of the IC2MAS architecture and the types of SAs this architecture integrates, we proposed four stages to handle the IC2MAS operating (cf. Figure 8): Initialization, Advertisement, Operation, and Maintenance. In what follows, the features of each stage are described. Note that Initialization and Advertisement stages are transparent to users.

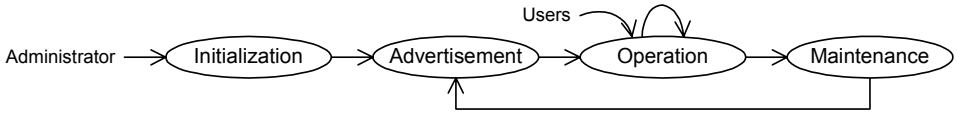

Figure 8 Stages of the IC2MAS operating

Initialization Stage - This stage is characterized by the following operations:

- The Advertisement Infrastructure and its components, i.e. Supervisor-Agent, Bulletin Board, and Repository of Active-Agents, are set up and started-up. Thus, the Supervisor-Agent initializes the Bulletin Board and the Repository. Further, this agent initiates the security policy that manages agents' accesses to the Advertisement Infrastructure.
- MASs are set up and associated with their respective CCISs. For instance, the Resolution-Agent creates its Help-Agent and sends it to the Advertisement Infrastructure (cf. Figure 9). As soon as it arrives, the Help-Agent is checked, registered, and finally, installed.

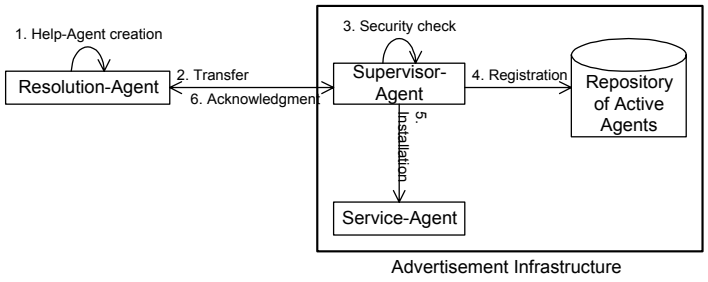

Figure 9 Help-Agent in the Advertisement Infrastructure

In what follows, we assume that, before leaving and entering MASs, Slave-Agents, namely Help-Agents and Route-Agents, interact with Control-Agents for security, shipping, and installation purposes.

Advertisement Stage - Once the initialization stage is done, CCIS-Agents have to advertise their services at the Advertisement-Infrastructure level. As stated in Section 3.3, CCIS-Agents send remote requests to the Supervisor-Agent of the Advertisement Infrastructure (cf. Figure 10).

According to the security level of this CCIS-Agent and the security policy of the Advertisement Infrastructure, the Supervisor-Agent decides if this CCIS-Agent is authorized to advertise and what types of services. In the positive case, the Supervisor-Agent processes the CCIS-Agent's request by posting the services it offers on the Bulletin Board. Furthermore, the Supervisor-Agent registers the fact that this CCIS-Agent has notes on the Bulletin Board. At the end, the Supervisor-Agent acknowledges the CCIS-Agent about the success (or failure) of the operation. We assume that CCIS-Agents send only one request in order to advertise all the services they offer. Moreover, we assume that other requests will follow that either update or withdraw the advertised services.

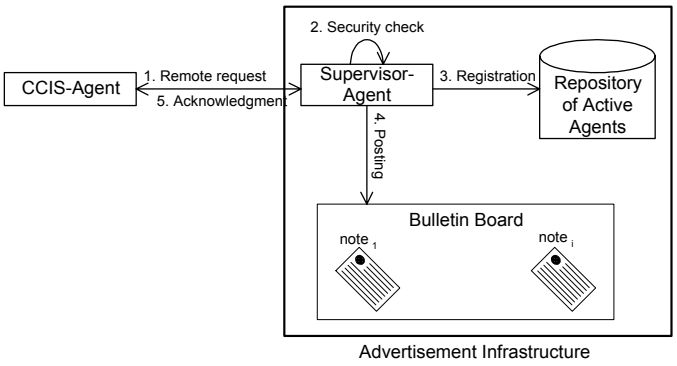

Figure 10 Services advertisement in the Bulletin Board

Operation Stage - Once the advertisement stage is done, the IC2MAS environment is ready to be operated. The operation stage of IC2MAS is summarized by two situations (cf. Figure 11):

- Only the user's CCIS is required: the CCIS-Agent is in charge of handling this situation (cf. Figure 11-a).
- Several CCISs, including or not the user's CCIS, are required: the Resolution-Agent is in charge of handling this situation (cf. Figure 11-b).

In what follows, numbers in parenthesis correspond to numbers in Figure 11 and illustrate operations chronology.

When a user wants to satisfy his needs (0), he interacts with the Interface-Agent of his MAS. Next, his needs are mapped into a request transmitted to the CCIS-Agent (1). This agent is in charge of deciding whether this user's CCIS contains the appropriate functions to process its request (cf. Figure 6, pre-processing module). Once such a decision is obtained (2), two situations exist and are identified in Figure 11 with letters a and b.

In Situation a, the CCIS-Agent forwards the user's request to the appropriate Function-Agent (3.a) of the user's CCIS. This Function-Agent initiates the CCIS's function and provides the results it obtained to the CCIS-Agent (4.a). Finally, results are sent to the user through the Interface-Agent (5.a, 6.a).

In Situation b, other CCISs, including or not the user's CCIS, are required to satisfy the user's request. These CCISs are identified using the notes of the Bulletin Board of the Advertisement Infrastructure. First, the CCIS-Agent forwards the user's request to the Resolution-Agent (3.b). Next, the Resolution-Agent interacts remotely with its Help-Agent, about the CCISs to identify (4.b). Once the Help-Agent has completed its operations (5.b), it sends to the Resolution-Agent the CCIS-Agents with whom it is going to interact (6.b). Once this information arrives, the Resolution-Agent starts to design its itinerary according to the number of the pertinent CCISs and the network status (7.b). To perform this itinerary, the Resolution-Agent creates a Route-Agent and assigns to this agent the designed itinerary (8.b). To clarify things, hereafter is an example illustrating this itinerary. In Figure 11, the itinerary indicates that the Route-Agent first has to move to a MAS (9.b), for instance MAS_2. Next, the Route-Agent interacts locally with the CCIS-Agent of this MAS (10.b). Furthermore, to complete its operations, the itinerary mentions that the Route-Agent has to remotely interact with other CCIS-Agents, for instance $CCIS\text{-}Agent_3$ of MAS_3. Then, the Route-Agent sends a request (11.b) and waits for the results from $CCIS\text{-}Agent_3$ (12.b). At the end, the Route-Agent goes back to its original MAS (13.b) and interacts with Resolution-Agent parent. Finally, the Resolution-Agent sends the results obtained from its Route-Agent to the user through the CCIS-Agent and the Interface-Agent (14.b, 15.b, 16.b).

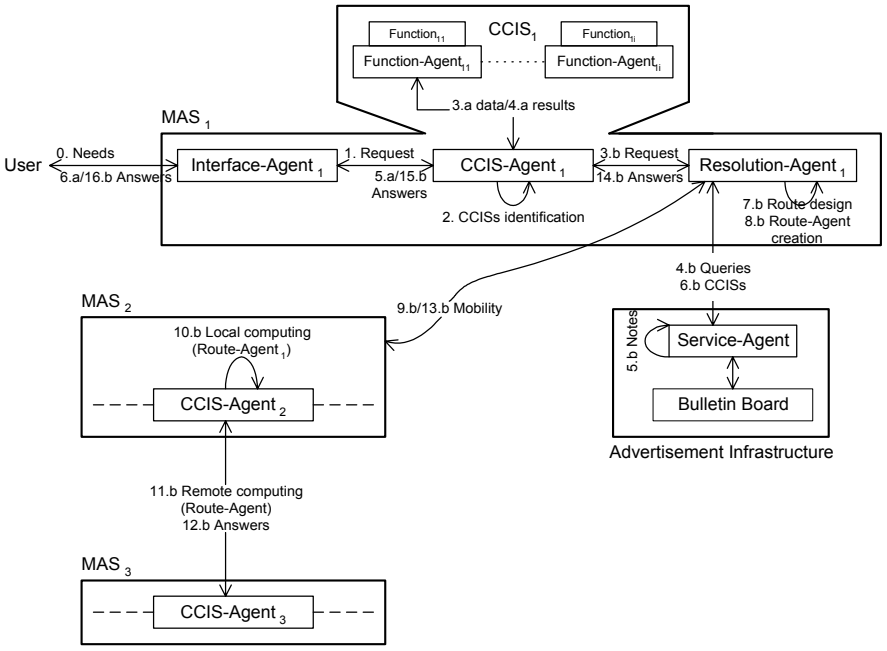

Figure 11 User's request satisfaction

Maintenance Stage - The IC2MAS environment is an open system. Indeed, a new CCIS could be integrated, another CCIS could be removed, etc. Therefore, the purpose of the maintenance stage is to take into account the situations that may have an impact on the architecture of the IC2MAS environment as well as on its operating. Several situations have been identified. In this paper, we briefly present two of them:

- It happens that a CCIS adapts its structural as well as functional characteristics, for example by adding a new function or by upgrading the version of a function's database management

system. Therefore, the CCIS-Agent has to be adapted either by adding new services to its capabilities or by updating its services. Further, the CCIS-Agent has to interact with the Advertisement Infrastructure.

- It happens that the Supervisor-Agent cleans up the Bulletin Board of the Advertisement Infrastructure, because of for example a new security policy. Hence, CCIS-Agents have to advertise their services from the beginning.

4. Related Work

This section summarizes the main characteristics of the IC2MAS environment with respect to other similar works. There exist different research projects in the field of systems interoperability [Bayardo et al. 1997] [Chawathe et al. 1994] [Genesereth and Ketchpel, 1994] [Hsu, 1996] [Knoblock et al., 1997] [Maamar et al., 1999] [Papazoglou et al., 1992]. All these projects have the same concerns, namely:

- Maintain the autonomy and independence of the systems to be integrated in an interoperable environment. In the IC2MAS environment, each CCIS has been associated with a CCIS-Agent that acts on its behalf.
- Reduce the informational disparities between the integrated systems. In the IC2MAS environment, the definition of an ontology is, currently, tackled (cf. Section 5.1).
- Help users satisfy their needs. In the IC2MAS environment, each MAS integrates an Interface-Agent that assists users.

However, all the projects cited above assume that the network infrastructure is fully reliable and has unlimited bandwidth for information transmission. In a coalition, this is not the case. In the IC2MAS environment, network concern has been considered, for instance by enhancing certain agents with mobility mechanisms and giving these agents the ability to decide whether local computing after a move is preferable than remote computing. Furthermore, security issues have been considered in the IC2MAS environment, by suggesting a security policy to manage the Advertisement Infrastructure and a security level to identify agents. Additional security elements could be suggested, for instance identifying services with authorization levels and users with use levels.

5. IC2MAS's Current Efforts

This section gives insights on topics that are currently, tackled, in the IC2MAS environment. Among these topics, we describe, briefly, the ontological disparities. Ontology is one of the main issues to be addressed in the design of an interoperable environment for heterogeneous systems. We consider an ontology as a means to represent and exchange information that are understood by all participants.

In a coalition context, each country has its own standards. Therefore, each military user specifies his needs, in term of information requests, and his CCIS's capabilities, in term of functions}, using these standards. Therefore, the need to define two types of specification languages is raised in the IC2MAS interoperability approach. The first type is a specification language for users' needs while the second type is a specification language for CCISs' functions. Both of these languages have to be based on two different ontologies: a user-oriented ontology and a CCIS-oriented ontology. Furthermore, because of the coalition context, the user-oriented ontology has to be adapted in order to take into account the individual differences, for example diversity of cultures that exist between the coalition's participants. To handle these characteristics, we intend to propose a user-oriented ontology that is "versioned" (certain authors talk about ontology sharing). Hence, only one user-oriented ontology is defined at the conceptual level but different versions of this ontology are defined at users level.

6. Conclusion

In this paper, we presented the major characteristics of the IC2MAS interoperability approach that uses MASs in the design of collaborative environments for distributed and heterogeneous CCISs. The coalition context is the running scenario. In this approach, MASs and their SAs are able to fulfill different operations, from users' needs specification to CCISs' functions initiation. Eight types of SAs exist in the architecture proposed for coalition support (Interface-Agent, CCIS-Agent, Resolution-Agent, Control-Agent, Function-Agent, Supervisor-Agent, Help-Agent, Route-Agent) while four stages describe this architecture operating (Initialization, Advertisement, Operation, Maintenance). Whereas MASs appear to

offer benefits to coalition support, we must be aware of their limitations. MASs must allow a large degree of human interaction. Therefore, it is unrealistic to expect to be able to provide a "fully" automated coalition support. A whole set of negotiations, dialogues, coordination and communication between participants, groups of participants, and systems are involved.

References

[Au, 1999] T.A. Au. A primer of CORBA: A framework for distribution applications in defence. Technical report, DSTO-GD-0192, DSTO Electronics and Surveillance Research Lab, Salisbury, Australia, 1999.

[Bayardo et al., 1997] R. Bayardo, W. Bohrer, R. Brice, A. Cichocki, G. Fowler, A. Helai, V. Kashyap, T. Ksiezyk, G. Martin, M. Nodine, M. Rashid, M. Rusinkiewicz, R. Shea, C. Unnikrishnan, A. Unruh, and D. Woelk. Infosleuth: Semantic integration of information in open and dynamic environments. In *Proc. of the 1997 ACM International Conference on the Management of Data (SIGMOD)*, Tucson, Arizona, 1997.

[Babin et al., 1994] G. Babin, C. Hsu, and Z. Maamar. A distributed metadatabase architecture for an enterprise scalable, adaptive integration environment. Technical Report 37-94-424, DSES, Rensselaer Polytechnic Institute, Troy, N.Y., USA, 1994.

[Bredin et al., 1999] J. Bredin, D. Kotz, and D. Rus. Mobile-agent planning in a market-oriented environment. Technical report, Dartmouth Technical Report PCS-TR99-345, 1999.

[Buschmann et al., 1996] F. Buschmann, R. Meunier, H. Rohnet, P. Sommerland, and M. Stal. *Pattern-Oriented Software Architecture*. John Wiley and Sons Ltd., 1996.

[Chawathe et al. 199] S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom. The TSIMMIS project: Integration of heterogeneous information sources. In *Proc. of the 10th IPSJ*, Tokyo, Japan, 1994.

[Fischmeister and Lugmayr, 1999] S. Fischmeister and W. Lugmayr. The supervisor-worker pattern. In *Proc. of Pattern Languages of Programs (PLoP'99)*, Kubova Hut, Czech Republic, 1999.

[Ghenniwa, 1997] H. Ghenniwa. *Coordination in Cooperative Distributed Systems*. PhD thesis, Systems Design Engineering Department, University of Waterloo, Ontario, Canada, 1997.

[Green et al., 1997] S. Green, L. Hurst, B. Nangle, P. Cunningham, F. Somers, and R. Evans. Software agents: A review. Technical report, Trinity College Dublin and Broadcom Éireann Research Ltd., May 1997.

[Genesereth and Ketchpel, 1994] M.R. Genesereth and A.P. Ketchpel. Software agents. *Communication of the ACM*, 37(7):48--53, July 1994.

[Hsu, 1996] C. Hsu. *Enterprise Integration and Modeling - the Metadatabase Approach*. Kluwer Academic Publisher, Boston, Mass., USA, 1996.

[Jones et al., 1998] D. Jones, T. Bench-Capon, and P. Visser. Methodologies for ontology development. In *the proceedings of IT&KNOWS - Information Technology and Knowledge Systems, 15th IFIP World Computer Congress*. Vienna (Austria) and Budapest (Bulgaria), 1998.

[Knoblock et al., 1997] C.A. Knoblock, Y. Arens, and C.N. Hsu. Cooperating agents for information retrieval. In *Second International Conference on Cooperative Information Systems*, Toronto, Canada, 1994.

[Labrou et al. 1999a] Y. Labrou, T. Finin, and Y. Peng. Agent communication languages: The current landscape. *IEEE Intelligent Systems*, 14(2):45--52, March/April 1999.

[Labrou et al. 1999b] Y. Labrou, T. Finin, and Y. Peng. The interoperability problem: Bringing together mobile agents and agent communication languages. In *Proceedings of the Hawaii International Conference On System Sciences*, Maui, Hawaii, January 1999.

[Lange and Oshima, 1999] D. Lange and M. Oshima. Dispatch your agents; shut off your machine. *Communication of the ACM*, 42(3):88--89, March 1999.

[Malerud et al., xxxx] S. Malerud, Feet E.H., and U. Thorsen. A method for analysing command and control systems. Norwegian Defence Research Establishment (FFI) N-220 Kjeller, Norway.

[Maamar et al., 1999] Z. Maamar, B. Moulin, and Y. Bédard. Software agent-oriented frameworks for the interoperability of georeferenced digital libraries on the world wide web: The SIGAL project. In R. Fegeas M.F. Goodchild, M.J. Egenhofer and C.A. Kottman, editors, *Interoperating Geographic Information Systems*, pages 335--354. Boston: Kluwer Academic Publishers, 1999.

[Papazoglou et al., 1992] M.P. Papazoglou, Laufmann S.C., and T.K. Sellis. An organizational framework for cooperating intelligent information systems. *International Journal of Intelligent and Cooperative Information Systems*, 1(1):169--202, 1992.

[Sycara et al. 1996] K. Sycara, K. Decker, A. Pannu, M. Williamson, and D. Zeng. Distributed intelligent agents. *IEEE Expert-Intelligent Systems and Their Applications*, 11(6):36--45, July 1996.