# Information & Security, Volume 8, Number 2

## Coalition Operations Planning and Negotiations



## Agents in Resource Planning



## I&S Monitor

### *I&S Research Centers*

Authors: **Zakaria Maamar, Nabil Sahli, Bernard Moulin, and Paul Labbé**
Title: **A Software Agent-based Collaborative Approach for Humanitarian-Assistance Scenarios**
Year of issuance: **2002**
Issue: **Information & Security. Volume 8, Number 2, 2002, pages 135-155**
Hard copy: **ISSN 1311-1493**

---

# A SOFTWARE AGENT-BASED COLLABORATIVE APPROACH FOR HUMANITARIAN-ASSISTANCE SCENARIOS

Zakaria MAAMAR, Nabil SAHLI, Bernard MOULIN and Paul LABBÉ

---

**Table Of Contents:**



---

## 1. Introduction

The purpose of this paper is to present a software agent-based approach to the design and development of a collaborative environment. Such an environment is addressed to users who aim at working together, despite their differences. This approach is part of the Collaborative Partners (CP) project.

The recent development of information and communication technologies has resulted in the existence of several types of systems. At different moments, several users from different organizations and with different background have to interact in order to perform common operations. However, these users are in different places, which imposes a *distribution* constraint, rely on different systems, which places a *heterogeneity* constraint, and use different communication languages, which puts a *vocabulary* constraint. Such constraints hinder the work of these users. Indeed, such users have to adapt their behavior in order to understand each other. Therefore, it becomes appropriate to propose a collaborative environment that will free users from worrying about their differences. In the CP project, we aim at suggesting a software agent-based approach to the design and development of such an environment. Collaboration relies on two elements: a coordination strategy and a communication language. In this paper, we focus on coordination.

The motivation behind the development of a collaborative environment is to promote the exchange of information and services. To this end, we suggest the use of two main concepts: Software Agents (SAs)[1] and coordination.[2] SAs have the ability to support the functionality of a collaborative environment, for example looking for relevant services on behalf of users, while coordination allows these agents to avoid conflicts. Furthermore, given distribution and heterogeneity constraints, mobility [3] and ontology [4] are key issues that are considered in the CP project. Mobility allows agents to migrate from one system to another to meet other agents and, hence, to coordinate their activities locally, while ontology resolves informational disparities that exist between users. By committing to an ontology, users of that ontology – either human or software agents – agree to represent their conceptualization of the world using a common language. In this paper, mobility and ontology are not discussed.

In the CP project, the software agent-based approach for collaboration is applied to humanitarian-assistance scenarios. In these scenarios, different nations, the United Nations (UN), and Non-Governmental Organizations (NGOs), e.g. The International Committee of the Red Cross and Médecins sans Frontières, have to work together in order to relieve the consequences of natural disasters. In any multinational operation, coordination is critical and represents shared responsibilities (cf. Figure 1). The term responsibility denotes a work for which a participant will be responsible. In addition, one of the most challenging issues in managing any multinational operation is organizing the different participants and structuring the coordination among them.[5] In fact, each participant has its own methods and priorities. It is stated that the ability to function successfully in a humanitarian assistance scenario depends on the capacity to function together.[6] In the CP project, we aim at developing a collaborative environment that will help the different participants in humanitarian scenarios to conduct their operations successfully. This environment will be described in the context of medical evacuation.

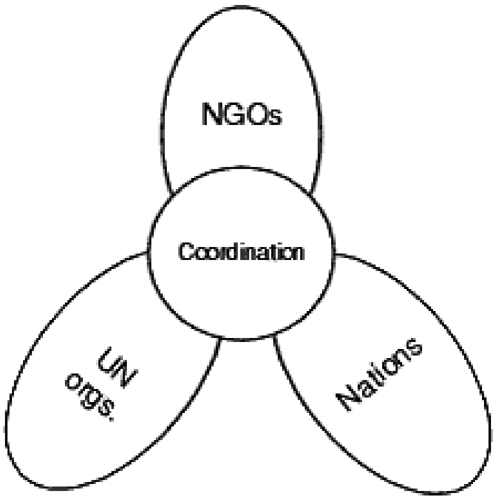



**Figure 1:** Coordination as shared responsibility

The paper is organized as follows. Section 1 proposes an overview of our theoretical, collaboration, and practical, humanitarian assistance, study. Section 2 presents the basic concepts that are used in this study. Section 3 describes the CP agent-based approach for collaboration and how it fits humanitarian-assistance scenarios. Section 4 deals with the medical evacuation domain as an illustration of humanitarian-assistance scenarios. Finally, Section 5 consists of concluding remarks.

## 2. Background

### *2.1. Humanitarian Assistance*

Nations, the UN, and NGOs are three major providers of humanitarian assistance in today's international community. Their methods of offering assistance and expected results may differ, but their general aim to provide relief to suffering is the same.[7] By consolidating each group's efforts, humanitarian assistance can become a more effective mission through the use of appropriate facilities, such as coordination.

Unfortunately, international humanitarian-assistance is not based on a well defined legal framework of laws, principles, or norms. Instead, the assistance is based on a set of informal arrangements. Therefore, there is no single structure which is applied to every assistance situation. In humanitarian scenarios, certain participants are better equipped and prepared than others. Usually, NGOs are considered as low-tech partners and rely on the services provided by the armed forces of other participants.

### *2.2. Coordination model*

Could we ensure efficient and effective interactions? This question has been raised many times. According to Tolksdorf, although a huge variety of coordination models exists, there is no common definition of the existing relations between coordination, communication, and cooperation.[8] In this section we provide a summary of the coordination models that has been studied in the CP project. Tolksdorf provides also an overview of coordination models.

**Naive model –** a basic understanding suggests a set of active entities working together to achieve common goals. It is assumed that these entities are willing to cooperate, for example, by sharing goals and exchanging information on these goals.

**Mintzberg model –** it is mainly used in organizations and relies on five mechanisms to achieve coordination. These mechanisms are mutual adjustment, direct supervision, work standardization, output standardization, and skill standardization. According to Mintzberg, the choice of coordination mechanisms is influenced by organizational structure. Furthermore, Mintzberg claimed that organizations are defined around five basic parts: strategic apex (overall responsibility), middle line (line of authority), operating core (work performance), support staff (that supports work indirectly), and techno-structure (where work is analyzed and planned).

**Coordination theory model –** it consists of dependencies that involve different components, namely goals, tasks, and resources. Yu extended these components; he proposed dependencies at the soft-goal level.[9]

## 3. Theoretical Perspective: The CP Approach

The CP agent-based approach for collaboration consists of three connected blocks (cf. Figure 2): organization, coordination, and operating block. *Organization* deals with the specification of an architecture that illustrates the participants of a humanitarian-assistance scenario. *Coordination* deals with the specification of the responsibilities in this scenario and the assignment of these responsibilities to participants. Finally, the *operating* block deals with the completion of these responsibilities, assessment of this performance, and how does this assessment influence the organization and coordination decisions, previously made. In Figure 2, the dashed line from the organization block to the operating block represents a setting-up process. In the same figure, the dotted line from the operating block to the organization block represents an adaptability process.

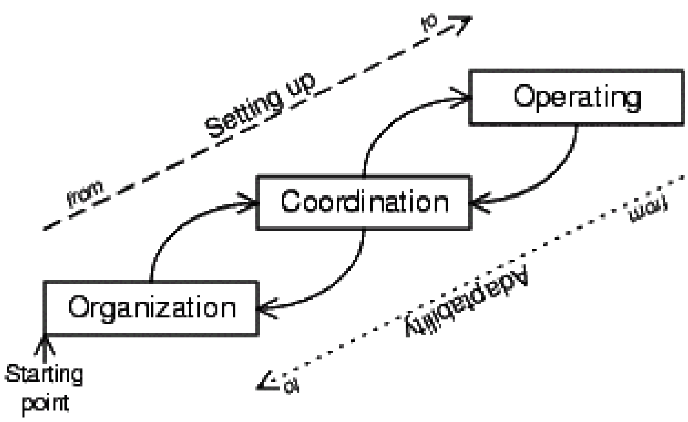



**Figure 2:** Building blocks of the CP Approach

### *3.1. Organization*

Figure 3 suggests an agent-based organization for humanitarian-assistance scenarios. This organization consists of two levels: commandment and operational level. The commandment level represents the structure that leads the accomplishment of the scenario. The operational level is in charge of carrying out the operations the commandment level prepares in terms of plans; it also assigns them. The operational level consists of all participants that have decided to contribute to the implementation of the humanitarian assistance scenario. In the example below, we assume that five participants are present on the field: two NGOs, one UN organization, and two nations.

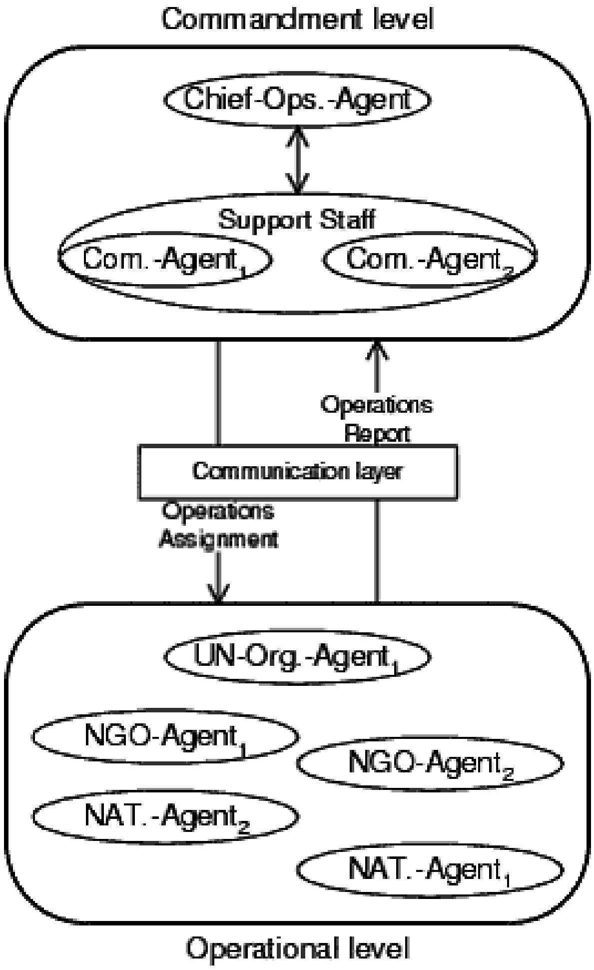



**Figure 3:** Agent-based organization for humanitarian scenarios

Generally, the commandment level has a chief of operations and support staff. This staff may include a commander for ground troops and a commander for civil affairs. The chief of operations interacts with the operational level through the support staff and *vice-versa*. It is important to note that even under unified command, the participants of the assistance scenario have different views and objectives.

At the beginning of a humanitarian-assistance scenario, the commandment level and the different components of the operational level are generally located in different regions of the country in which the scenario occurs. Furthermore, after a disaster the support facilities of a recipient country, such as communication infrastructure, are usually destroyed or not fully reliable. Therefore, the commandment and operational levels communicate through devices that have low-bandwidth and/or unreliable channels, for example VHF Combat Net Radios. The commandment level assigns to the operational level the operations they have to undertake. The operational level carries out these operations and reports their progress to the commandment level. Reports allow assessing the situation for adjustment purposes.

We have decided in the CP project that certain responsibilities of the participants in the assistance-scenario are going to be delegated to agents. An agent can be launched to monitor the events (such as risk of showers) that are of interest to an user and notify him, when needed. Moreover, an agent can support the Chief of operations in his daily duties, e.g. filtering reports. In Figure 3, the commandment level integrates three types of agents and the operational level integrates five types of agents (cf. Table 1). In this Table, NAT.-Agent $_1$ is aware of the doctrines and policies that rule out the behavior of its forces in different situations, such as participation in coalition operations.

**Table 1:** Agent types and designation

| Type | Designation |
|---|---|
| Chief-Ops.-Agent | Chief of operations |
| Com.-Agent $_1$ | Commander of ground troops |
| Com.-Agent $_2$ | Commander of civil affairs |
| UN-Org.-Agent $_1$ | UN organization |
| NGO-Agent $_1$ | NGO #1 |
| NGO-Agent $_2$ | NGO #2 |
| NAT.-Agent $_1$ | Nation #1 |
| NAT.-Agent $_2$ | Nation #2 |

### *3.2. Coordination*

In the CP approach, coordination is among the responsibilities of the commandment level. It is used to identify the four Ws, namely Who, What, When, and Where. The Chief-Ops.-Agent assigns and distributes the responsibilities to the participants in the humanitarian-assistance scenario. For instance, Forces of Nation $_1$ are given the task to provide food and medical care. Assigning responsibilities is not a static process; assignments may change to reflect the capabilities of participants during the progress of operations.

In reality, the participants, particularly the nations with their deployed armed forces, have their doctrines. Therefore, coordination aims to avoid conflicts between doctrines and solves problems that prevent certain operations from being performed successfully. Conflicts occur at different levels: equipment and logistics, training and doctrines, intelligence, languages and cultures.[10] Coordination has also an impact on the assignment of responsibilities. As stated in Section 2.1, certain participants are not well equipped due to specific shortfalls, e.g. lack of computers. Therefore, coordination allows associating these participants with other well-equipped participants in order to work under their supervision and, thus, to use their resources.

Figure 4 illustrates how coordination can be achieved in humanitarian-assistance scenarios. In the same figure, new types of agents, called Delegate-Agents, are introduced. Each Delegate-Agent corresponds to an agent of the operational level. For example, Delegate-Agent $_1$ advertises to the Chief-Ops.-Agent the capabilities (e.g. providing medical care), requirements (e.g. transporting supplies by air), and doctrines (e.g. no weapons in assistance scenarios) of UN-Org.-Agent $_1$. Advertisements are made in a way that the Chief-Ops.-Agent and its support staff can understand.

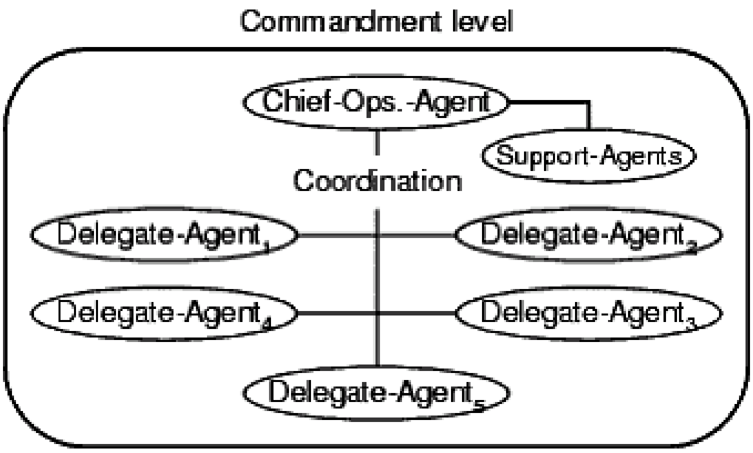



**Figure 4:** Agent-based coordination for humanitarian scenarios

Coordination is either implicit or explicit. Implicit coordination assumes that agents follow specific rules. These rules are predefined and related to a specific application domain, for example car traffic. In humanitarian scenarios, implicit coordination is part of the doctrines of armed forces. Within different units of the same country, a soldier obeys his hierarchical chief. However, if the chief is from another country, the soldier can refuse to obey. Such situation requires an explicit coordination. To this end, agents use different policies, such as negotiation, voting, or intervention of an authority. In humanitarian scenarios, the Chief of Operations represents the authority. He sets up the rules that dictate the behavior of every participant.

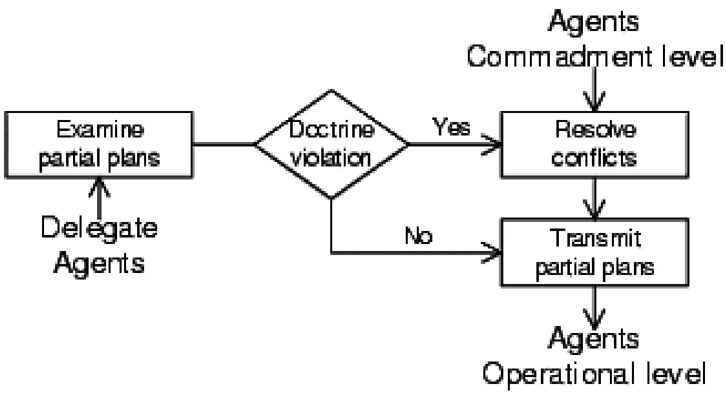



**Figure 5:** Conflict resolution between partial plans

In Figure 4, the coordination strategy is set up as follows. First, Delegate-Agents introduce themselves to the Chief-Ops.-Agent and its Support-Agents. Next, all the agents collaborate to specify the operations each member of the operational level has to perform. Operations are packaged into plans that are partial and conflicts-free. Plans are incomplete and the members of the operational level have to detail them according to the situations in which they are involved. For example, the Forces of Nation $_1$ can be tasked to build a camp. The partial-plan of building a camp only specifies the number of refugees to expect and the location of the camp. Plans are conflicts-free because they do not compete against each other at the resource level; the commandment level has taken into account the situations that could lead to conflicts among the members of the operational level. Moreover, Delegate-Agents are involved in defining the partial plans. However, when the agents of the operational-level implement the partial plans, these plans become global (i.e. from partial plans to global plans). According to the previous example, in order to build a camp Nation $_1$'s Forces should move to the camp's location, gather appropriate supplies, build the camp, provide the camp with basic furniture, and wait for the arrival of refugees. All these activities constitute the global plan of building a camp. Within and between global plans, however, conflict situations could occur that require the intervention of the commandment level (refer to Figure 5 which is adapted from the work of Liu and Sycara [11]). Table 2 summarizes the main elements of coordination in humanitarian-assistance scenarios.

**Table 2:** Coordination elements in humanitarian-assistance scenarios

| | **Commandment level** | **Operational level** |
|---|---|---|
| Central Planning | Yes | N/A |
| Distributed Planning | N/A | Yes |
| Result | Partial Plans | Global Plans |
| Is coordination required? | No | Yes |

**Definitions**

Delegate-Agent = Advertise(**C***apacity*, **R***equirement*, **D***octrine*) to Commandment level

**Capacity C** – a set of capacities to advertise
**C** = $\{C_1, C_2, \ldots, C_i\}$; $C_i$ - provide medical care;

**Requirement R** – a set of requirements to satisfy
**R** = $\{R_1, R_2, \ldots, R_j\}$; $R_j$ - air transportation;

**Doctrine D** – a set of rules that dictate how to behave
**D** = $\{D_1, D_2, \ldots, D_k\}$; $D_k$ - no weapons in an assistance scenario.

**Constraints**
Assign: {Operations} $\rightarrow$ {Participants}
$Op_j \rightarrow P_k$

- contradict($P_k \bullet D_i$, $Op_j$): Operation $Op_j$ does not contradict Doctrine $D_i$ of Participant $P_k$.
- able($P_k \bullet C_i$, $Op_j$): with Capacity $C_i$, Participant $P_k$ is able to perform Operation $Op_j$.
- need($P_k \bullet R_i$, $Op_j$): Participant $P_k$ needs the Requirement $R_i$ to perform Operation $Op_j$.

### *3.3. Operating block*

Figure 6 illustrates the operation of an agent-based system that is applied to humanitarian-assistance scenarios. Once the commandment level defines the partial plans to be performed, and also the coordination strategy to be followed, Delegate-Agents interact with their respective agents of the operational level. For instance, the commandment level asks certain participants to work under the supervision of other participants. In Figure 6, three groups are set up: Group $_1$ consists of UN-Org.-Agent $_1$; Group $_2$ consists of NAT.-Agent $_1$ and NGO-Agent $_2$; and, Group $_3$ consists of NAT.-Agent $_2$ and NGO-Agent $_1$. In this example, NAT.-Agent $_1$ supervises NGO-Agent $_2$, i.e. Médecins sans Frontières. Therefore, NGO-Agent $_2$ uses the facilities that are provided by the Forces of Nation $_1$.

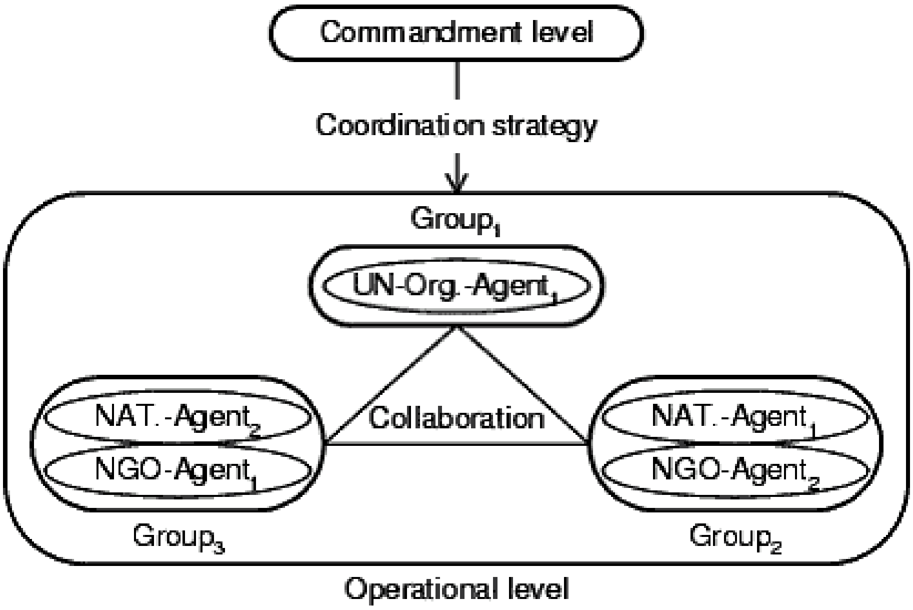



**Figure 6:** Agent-based system operating in humanitarian assistance scenarios

Once the agents that lead the different groups, i.e. UN-Org.-Agent $_1$, NAT.-Agent $_1$, and NAT.-Agent $_2$, receive the partial plans, they start detailing these plans according to their doctrines and the situations in which they are involved. For instance, NAT.-Agent $_1$ generates global plans and assigns part of them to NGO-Agent $_2$. Once the global plans are carried out, reports of the progress of ongoing operations are transmitted to the commandment level for follow-up purposes. In the building camp scenario, Forces of Nation $_1$ ask Médecins sans Frontières to supply the camp's clinic with medical facilities. Then, Forces of Nation $_1$ inform a member of the commandment level that the camp is completed and ready to receive refugees.

In the CP approach, a global plan consists of the following elements:

**What** is the goal of the global plan?
Example: to build a camp for refugees.

**What** are the operations to be performed?
Example: move to the camp location; gather appropriate supplies; build the camp; provide the camp with basic furniture;

**Where** to perform the operations?
Example: zone $_1$ – to build the camp; zone $_2$ – to get supplies; zone $_4$ – to get furniture.

**When** to carry out the operations?
Example: move to the camp location on January 26th; gather appropriate supplies on arrival to zone $_1$.

**What** are the required resources for operations?
Example: lorries; personnel; air transportation.

**Who** performs the operations?
Example: Nation $_1$'s Forces for Ops $_{1,i,n}$; Médecins sans Frontières for Ops $_{i+1,\, n-1}$.

At the operational level, it may happen that a group of agents cannot execute an activity of a global plan on their own, i.e. they lack resources. Thus, they request support from other agents. To identify appropriate agents, these agents interact with the commandment level, through their Delegate-Agents. For example, during gathering the needed supplies Forces of Nation $_1$ ask Forces of Nation $_2$ for air transportation due to floods. Before Forces of Nation $_2$ commit themselves to provide air transportation, their NAT.-Agent $_2$ schedules the transportation request and informs the commandment level about its commitment.

In case when a global plan contradicts the doctrine of a participant of the operational level, the Chief of operations is informed (cf. Figure 7). For example, an NGO has been requested to work in a risky area. However, the NGO neither has weapons to protect its members nor is allowed to carry weapons. Therefore, the NGO reports this situation to the commandment level. The Chief of Operations asks appropriate military forces to escort the members of the NGO.

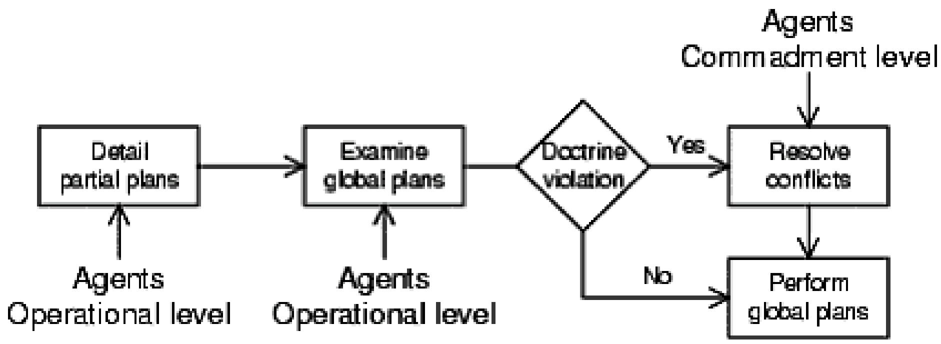



**Figure 7:** Conflicts resolution in global plans

### *3.4. Summary*

The organization, coordination and operating blocks of the CP approach identify the steps that can achieve an efficient collaboration support based on software agents. First, users organize themselves according to the goals to be accomplished and to the characteristics of their working environment, such as cultural and political orientation. Next, users understand each other intentions. Therefore, they agree on what to do, who is going to do it, and what to expect from each other. Finally, users carry out their work and assess the outcomes for improvements and adjustments. The CP approach assumes that users delegate part of their work to agents. In this way, users can spend more effort on other aspects, such as defining strategies for collaboration.

## 4. Application Perspective: Medical Evacuation

In this section, we discuss a situation that usually occurs in humanitarian-assistance scenarios. Medical evacuation denotes this situation and aims at transferring patients to the appropriate medical facilities, according to the condition of patients and the capabilities of facilities.[12]

### *4.1. Problem Statement*

In humanitarian-assistance situations, an intervention area is usually decomposed into several regions (cf. Figure 8). Each region has a commander in chief who is in charge of conducting operations. We associate regions with region-agents. These agents are aware of the incoming and outgoing flows occurring within their regions. In the rest of this paper, we focus on the flow involving patients. Each region has several Transportation Centers (TCs), two of them are used for incoming and outgoing flows, respectively. They can be both located either in the same place or in different places. Each TC is associated with a TC-agent. Usually, each region has several Medical Facilities (MFs). MFs are associated with MF-agents, for management purposes. For instance, a MF-agent knows how many unoccupied beds are available, what kind of expertise exists, and what are the TCs reaching a MF.

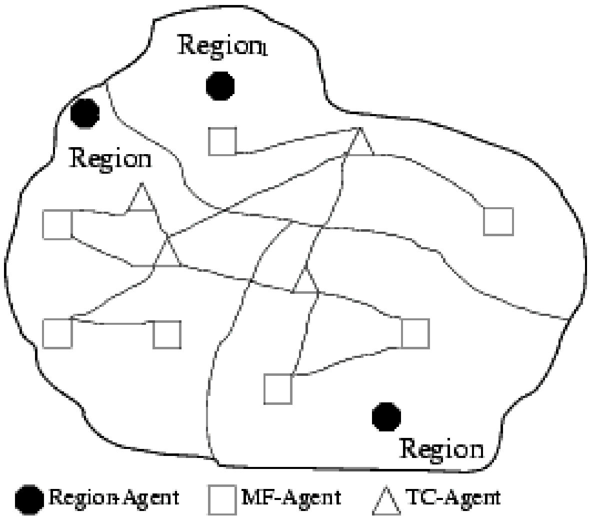



**Figure 8:** Agentification of intervention areas

It may occur that a patient requires specific treatment. As a first step, the patient is sent to the MF of his region. There, the patient receives the initial treatment, waiting for his transfer to an appropriate MF either in the same region (intra-regional move) or in another region (inter-regional move). Transferring patients between MFs is subject to the following constraints:

- The authority of each region, i.e. the commander in chief, is informed that other MFs intend to use the MFs and TCs of the region he is in charge of.
- Conflicts over resources, e.g. beds, can take place mainly in inter-regional moves.
- Finally, each patient needs an itinerary that specifies the initial MF, the final MF and the MFs and TCs this patient will visit during his transfer. MFs are constrained by the number of unoccupied beds and transport vehicles are limited by their number of available seats.

### *4.2. CP approach as a solution*

The application of the CP approach to the medical-evacuation case goes through two phases: planning and execution (cf. Figure 2). Planning is associated with organization and coordination blocks, while execution is associated with the operating block. Interleaving planning and execution is the main characteristics of the application of the CP approach to medical evacuation.

#### *4.2.1 Planning meets organization and coordination blocks*

In the planning phase, a MF that meets the requirements of a patient has to be identified. Each patient has a medical record that is managed by a patient-agent. First, the patient-agent sends messages to the region-agents, looking for the MFs that will be able to host its patient. Region-agents forward messages to their MF-agents. As soon as patient-agent starts receiving replies, the first choice is an MF that is within the region of the patient. The second choice is a MF in another region.

- First choice – intra-regional move: after identifying the MF that will host its patient, the patient-agent asks the MF-agent to reserve a bed for this patient. The MF-agent keeps this booking active, for a specified period of time. This period depends on the condition of the patient as well as on the booking rate in this MF. The MF-agent then sends a reply to the patient-agent with information on the booking together with an expiration date. The next step for the patient-agent consists of searching for the TCs that reach this MF. According to the number of TCs, a possible itinerary is designed and a confirmation about the arrival date of the patient is sent to the MF-agent.
- Second choice – inter-regions move: it may happen that the MFs of a region do not offer the treatment that a patient requires. Thus, the patient-agent of this patient looks for MFs in other regions. An inter-regions move requires transferring the patient:

- From the initial MF to the outgoing TC of the original region;

- From this outgoing TC to the incoming TC of the destination region;

- From this incoming TC to the selected MF.

As soon as a MF in another region is identified, the patient-agent proceeds as follows: it creates a delegate-agent and ships this delegate-agent to the outgoing TC of the patient's original region. The delegate-agent is in charge of planning the sub-itinerary of the patient from this TC to the final destination, which is the selected MF. Meanwhile the patient-agent is in charge of planning the sub-itinerary that leads the patient from the original MF to the outgoing TC. In fact, the patient-agent entrusts part of its responsibilities to the delegate-agent (cf. Figure 9).

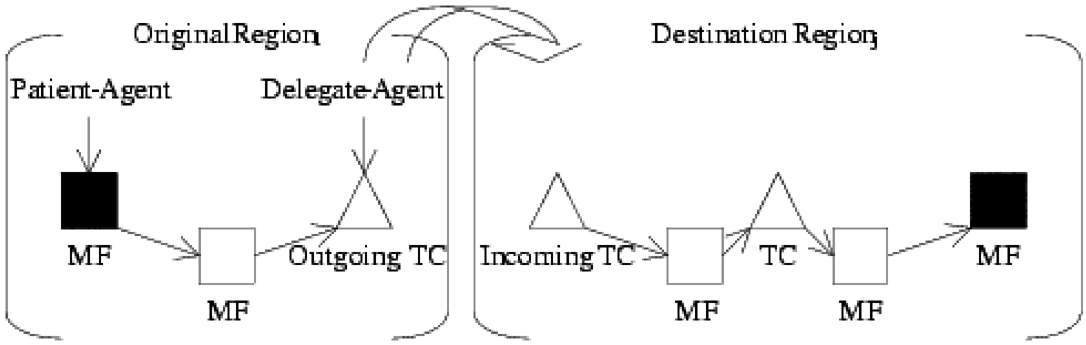



**Figure 9:** Patient-agent delegating its responsibilities

As soon as it arrives at destination, i.e. the outgoing TC, the delegate-agent queries the MF-agents, through region-agents, regarding the expertise of their MFs. To this end, the delegate-agent specifies the condition and requirement of the patient (we assume that at least one MF will answer positively). When it gets a reply from a specific MF, consisting of a booking with an expiration date, the delegate-agent interacts with the incoming TC of the region of this MF, regarding the itinerary that needs to be followed within that region. Finally, the delegate-agent confirms to the MF destination the arrival date of the patient. At the same time, the patient-agent works on establishing a sub-itinerary for its patient from the original MF to the outgoing TC. As soon as the patient-agent completes the sub-itinerary, the patient is transferred to this TC. The patient-agent is, also, part of the transfer. When the patient-agent arrives, the delegate-agent provides the patient-agent with the sub-itinerary that needs to be followed. Finally, the delegate-agent is deleted.

#### *4.2.2 Execution meets operating block*

Once its itinerary is established, the patient is evacuated to the destination MF. This evacuation consists of visiting several TCs and several MFs. However, each step of the itinerary can be subject to changes. Therefore, the patient-agent is attached to its patient. Dealing with any change requires from the patient-agent to create another delegate-agent. This agent will precede the patient-agent by one step. In fact, the role of the delegate-agent is to visit the destination site for a given step and notify its patient-agent in case unforeseen events occur. Examples of these events can be a delayed departure from a TC, a canceled departure, etc. In order to deal with these events, the delegate-agent needs to have information on the expected state of a site in order to compare it with the current state of this site. In case of any difference, the patient-agent is informed for corrective actions. To this end, the patient-agent needs to re-plan a part of the plan that deals with the difference.[13] For instance, if a departure to a TC has been canceled, it would be more appropriate to be aware of this situation before the patient arrives at this TC.

### *4.3. Implementation*

Figure 10 presents a snapshot of the system that is used for simulation of the medical evacuation scenario. In the system, agents are implemented as Java classes, using JDK 1.2.2 for Windows. Moreover, the mobility of certain agents is achieved using the middleware Beegent from Toshiba.[14] In Beegent, two types of agents exist, namely wrapper (static) and mediator (mobile).

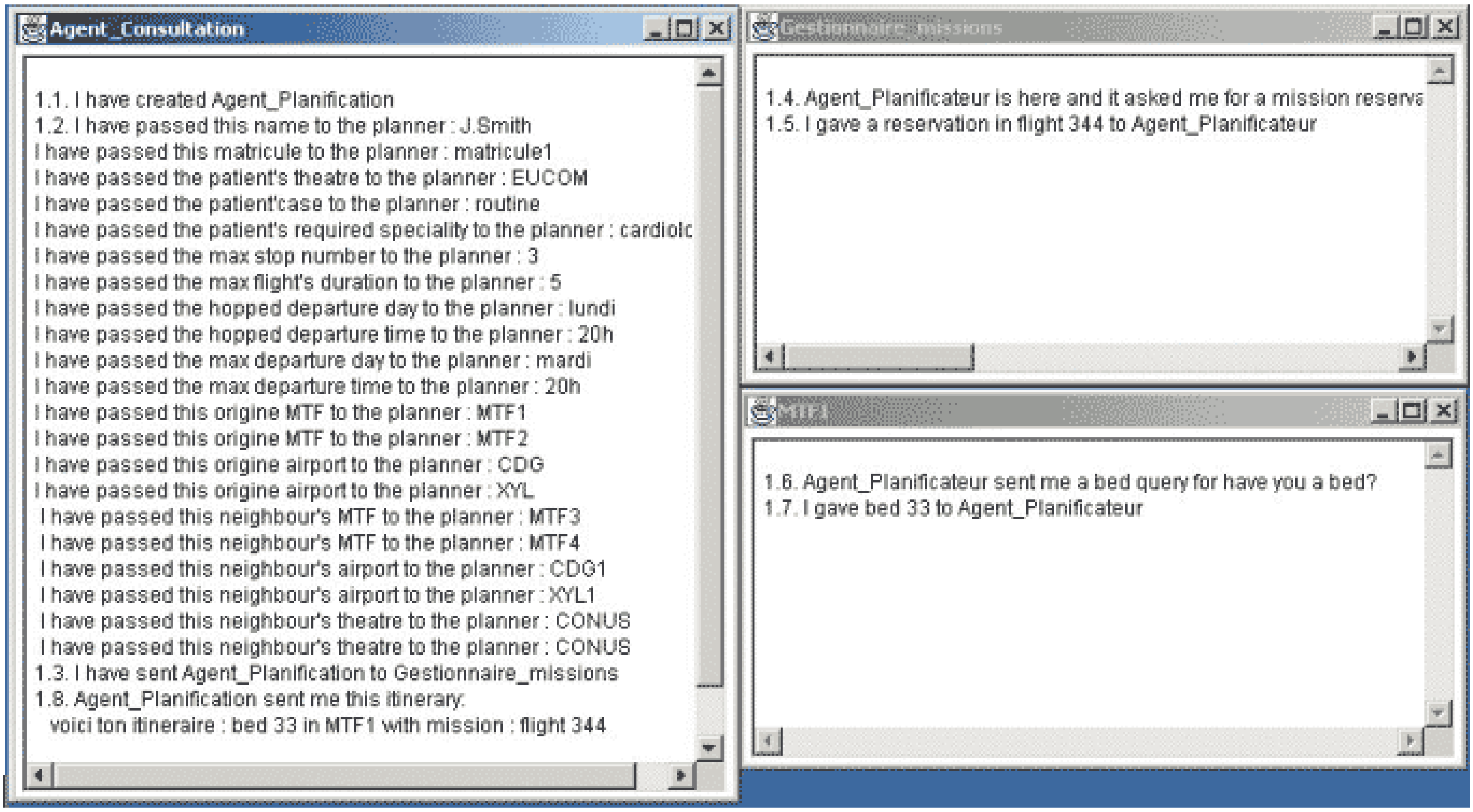



**Figure 10:** Snapshot of the medical evacuation system

Different agents have been implemented: region-agents associated with regions, TC-agents associated with TCs, MF-agents associated with MFs, and finally patient-agents associated with patients. Delegate-agents are dynamically created by patient-agents when needed. Region-agents, TC-agents, and MF-agents inherit from the wrapper class while patient-agents and delegate-agents inherit from the mediator class. Both agents may have to move from one region to another. Figure 10 is an example of the interactions that take place between a patient-agent, a region-agent and the MF-agent of $MF_1$. Different types of messages are exchanged between them and are related to the condition and requirements of the patient, unoccupied beds of $MF_1$, and next departures to this MF from the region of the agent. The result of all these messages is an itinerary assigned to the patient: *bed 33 of* $MF_1$ *with flight 344.*

### 4.4. Summary

In addition to organization, coordination, and operating blocks, the functioning of the CP approach is characterized by interleaved planning and execution phases. The traditional separation between planning and execution is no longer relevant, especially in dynamic environments such as humanitarian assistance. Agents need to adapt their configuration to consequences of the actions other agents carry out, as well as of the events that were unforeseen. Ambroszkiewicz claims that if there are several agents that may help or prevent an agent from achieving its goal, computing a plan may take so long that after completing the computation, the plan becomes out-of-date since the world has changed meanwhile.[15]

In the work of Paolucci, Shehory, and Sycara interleaving planning and execution means interrupting planning, executing the portion of the plan that has been designed, and finally, resuming planning. [16] This is time-consuming, particularly in critical situations. In the CP approach, the planning is not suspended at all; a delegate is created and mandated to pursue planning. As soon as the agent finishes its execution, it interacts with the delegate regarding the new portion of the plan to be carried out. Therefore, interleaving planning and execution is enhanced with delegation.

## 5. Conclusions

In this paper, we presented the major characteristics of the CP approach for collaboration support. This approach relies on software agents as a means to achieve this support. Furthermore, we presented how the CP approach meets the requirements of humanitarian-assistance scenarios. In such scenarios, different participants have to work together, despite their differences at operational, strategic, and cultural levels. In order to ensure an efficient collaboration among the participants, software agents turned out to be a good candidate. For instance, they collaborate on behalf of their users, monitor the events that are of interest to users, coordinate their activities and avoid conflicts. Therefore, users can concentrate on other activities, such as assessing situations, making decisions, or reacting to change. In this paper, we have also described the application of the CP approach to medical evacuation as a special case of humanitarian-assistance scenario. Delegation allowed interleaving planning and execution.

**Acknowledgement**: The medical evacuation project, described in Section 4, has been supported by contract No. W7701-8-4687 from the Defense Research Establishment Valcartier, Québec, Canada.

**Disclaimer**: The views and conclusions contained in this paper are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Zayed University, Laval University, or DRDC.

**Appendix 1**

Figure 11 is the interface that is used to get information on the patient's condition and requirements.

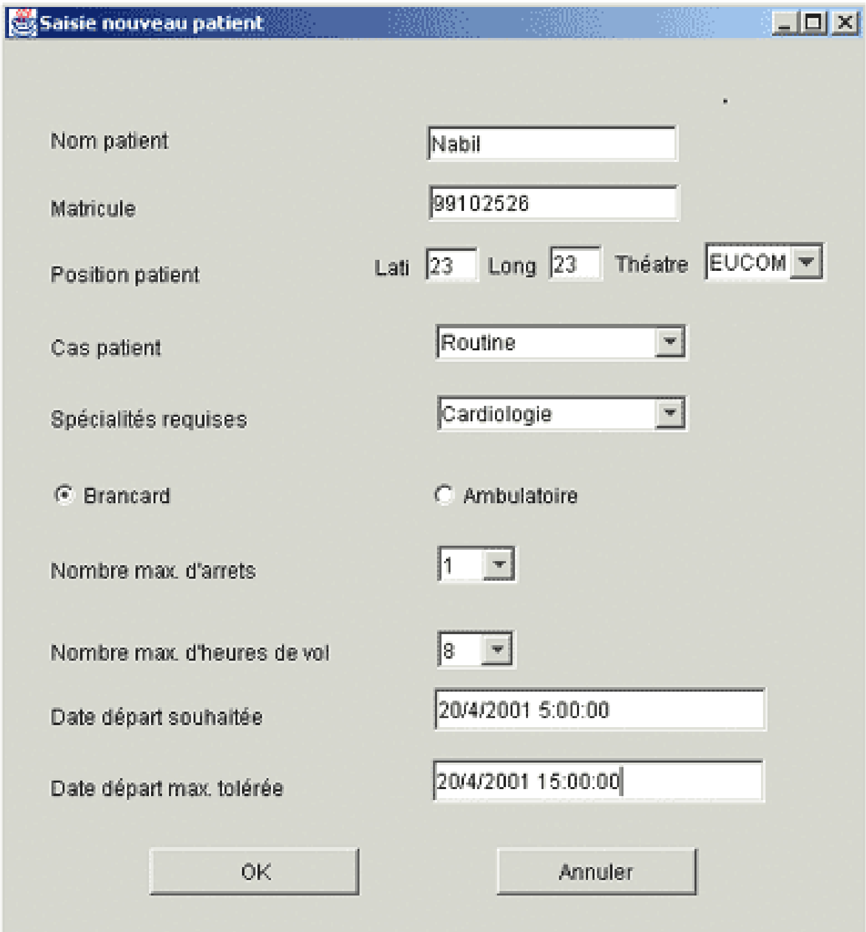



**Figure 11:** User-interface for the patient

**Appendix 2**

Figure 12 is the interface that is used for displaying the relevant MFs, according to the patient's condition and requirements.

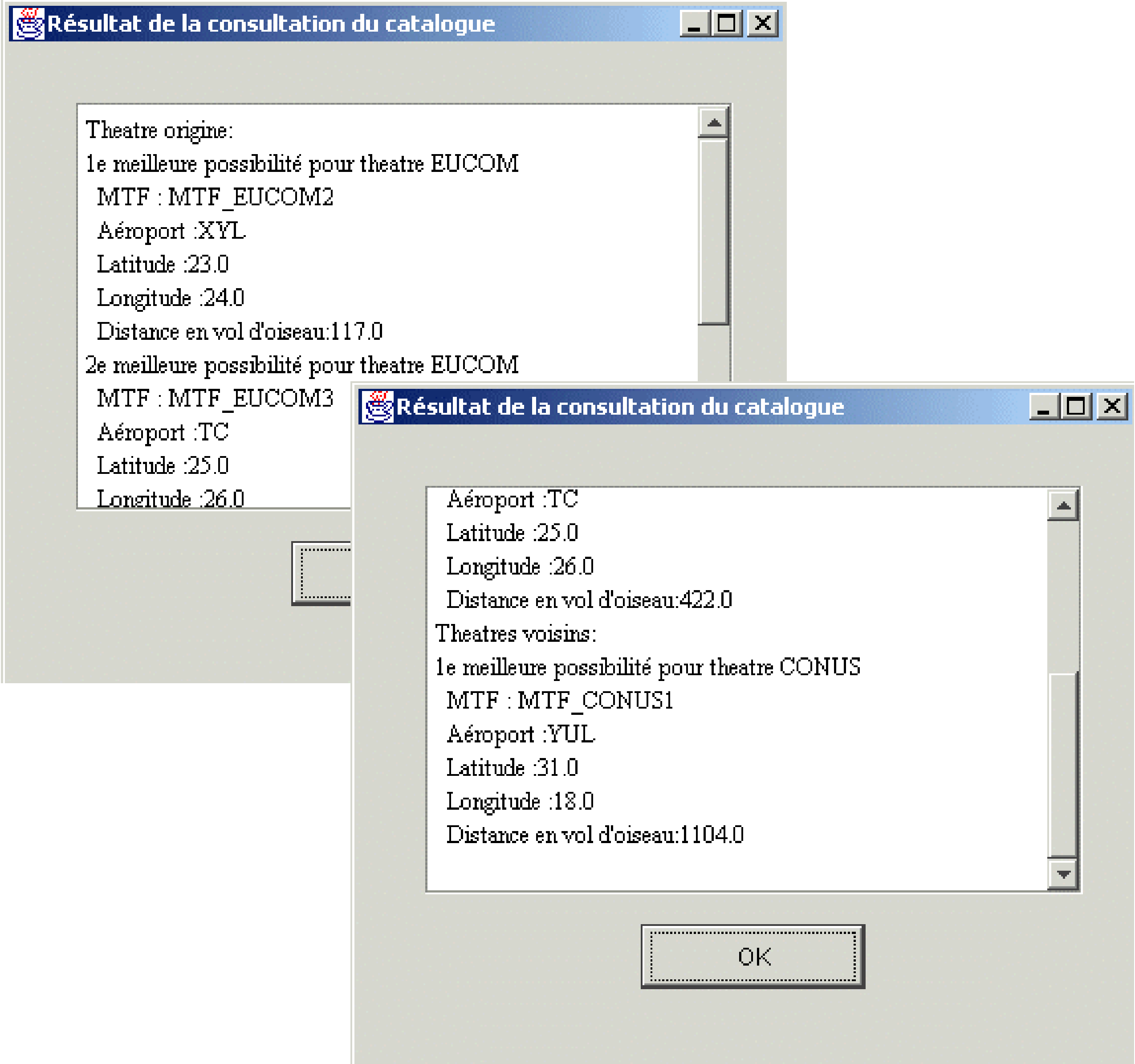



**Figure 12:** MFs presentation

---

**Notes:**


1. N. Jennings, K. Sycara, and M. Wooldridge, "A roadmap of agent research and development," *Autonomous Agents and Multi-Agent Systems* 1, no.1 (1998): 7-38.
2. T. Malone and K. Crowston, "The interdisciplinary study of coordination," *ACM Computing Surveys* 26, no.1 (1994): 87-119.
3. D. Lange and M. Oshima, "Dispatch your agents; shut off your machine," *Communication of the ACM* 42, no.3 (March 1999): 88-89.
4. *IJCAI-99 Workshop on Ontologies, Problem-Solving Methods: Lessons Learned, and Future Trends (1999)* <http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-18/> (21 March 2002).
5. H.E. Bullock, *Peace by committee: Command and Control issues in multinational peace enforcement operations*, Technical report (Maxwell AFB, Ala.: School of Advanced Airpower Studies, 1995).
6. A. Dayon, *Defining the relationship of the US government, United Nations, and non-governmental organizations in humanitarian assistance*, Master's thesis (Graduate School of Logisitics and Acquisition Management of the Air Force Institute of Technology, Air University, 1995).
7. Dayon, *Defining the relationship.*
8. R. Tolksdorf, "Models of coordination," in *Proceedings of the International Workshop on Engineering Societies in the Agents' World ESAW'00* (Italy, 2000).
9. E. Yu, "Strategic modelling for enterprise integration," in *Proceedings of the 14th World Congress of the International Federation of Automatic Control* (Beijing, China: IFAC, 1999).

10. Bullock, *Peace by Committee.*

11. J.S. Liu and K. Sycara, “Emergent constraint satisfaction through multi-agent coordinated interaction,” in *Proceedings of the Fifth European Workshop on Modeling Autonomous Agents in a Multi-Agent World* (1993).

12. V. Saks, G. Braidic, A. Kott, and C. Kirschner, “Distributed medical evacuation planning: What problem should each agent solve?” in *Proceedings of the Workshop on Constraints and Agents, Fourteenth National Conference on Artificial Intelligence AAAI'97* (Providence, Rhode Island, 1997); O. Lassila, M. Becker, and S. Smith, *An exploratory prototype for reactive management of aeromedical evacuation plans*, Technical Report CMU-RI-TR-96-03 (Pittsburgh: The Robotics Institute, Carnegie Mellon University, 1996).

13. V. Marik, M. Pechoucek, O. Stepankova, and J. Lazansky, “Proplant: Multi-agent system for production planning,” *International Journal of Applied Artificial Intelligence* (September 2000): suggest a classification of re-planning types. Re-planning aims at changing a plan that already exists. In addition, it mainly occurs when a system configuration is object to modifications, such as an agent acquiring a new capacity. Re-planning takes place at different stages: during the planning, after the plans are completed, or during the plans’ execution. Several reasons motivate re-planning and are sorted into two types: request-driven planning and community-driven planning. In the first type, the purpose is to integrate a high-priority plan. The execution of this plan requires postponing the plans that are either waiting for their execution or under execution. In the second type, the purpose is to deal with the modifications that occur in the agent community, such as an agent leaving the community and an agent losing a capacity.

14. *Beegent* <http://www2.toshiba.co.jp/beegent/index.htm> (15 December 2001).

15. S. Ambroszkiewicz, “Agent virtual organizations within the framework of network computing: a case study,” in *Proceedings of The First International Workshop of Central and Eastern Europe on Multi-agent Systems CEEMAS'99* (St. Petersburg, Russia, 1999).

16. M. Paolucci, O. Shehory, and K. Sycara, *Interleaving planning and execution in a multiagent team planning environment*, Technical report CMU-RI-TR-00-01 (Pittsburgh: The Robotics Institute, Carnegie Mellon Univeristy, 2000).

---

**ZAKARIA MAAMAR** is an associate professor at the College of Information Systems of Zayed University. In addition, he is respectively an adjunct professor at the Computer Sciences Department and an associate researcher at the Research Center in Geomatics, both of Laval University, Quebec, Canada. Dr. Maamar received two IEEE awards in 1999 and 2001, respectively. Dr. Maamar’s research interests are mainly related to software agents, wireless systems, and web and mobile services. Dr. Maamar has published several papers in journals and conference proceedings. Dr. Maamar graduated for his M.Sc. and Ph.D. in computer sciences from Laval University, Quebec, Canada in 1995 and 1998, respectively. *Address for correspondence:* Software Agents Research Group, College of Information Systems, Zayed University, PO Box 19282, Dubai, United Arab Emirates, E-mail: zakaria.maamar@zu.ac.ae; Research Center in Geomatics and Computer Science Department, Laval University, Ste-Foy QC G1K 7P4, Canada.

**NABIL SAHLI** is a Ph.D. student in computer science at Laval University, Quebec, Canada. He graduated for his engineering diploma in computer science in 1999. His research interests are related to stationary and mobile agents in solving complex problems within open and highly dynamic environments. *Address for correspondence:* Computer Science Department, Laval University, Ste-Foy QC G1K 7P4, Canada, E-mail: sahli@ift.ulaval.ca

**BERNARD MOULIN** is a full professor at Laval University, teaching at the Computer Science Department. He is also the head of the Research Center in Geomatics at Laval University. He received his engineering degree from L'Ecole Centrale de Lyon (1976), his Master in economics from the University Lyon II (1976) and his Ph.D. in computer sciences from the University Lyon I (1979) (all in France). He leads several research projects in various fields: design methods for multiagent systems and software-agent environments; representation of temporal and spatial knowledge in discourse. These projects are funded by several agencies, such as the Natural Science and Engineering Council of Canada, the research council FCAR from Quebec, and the Canadian Network of Centers of Excellence in Geomatics GEOIDE. *Address for correspondence:* Research Center in Geomatics and Computer Science Department, Laval University, Ste-Foy QC G1K 7P4, Canada, E-mail: moulin@ift.ulaval.ca

**PAUL LABBÉ** joined the National Defense Canada in 1978. While at the Communications Research Centre in Ottawa 1984-89, he was responsible for the research activities on naval communications architectures and data network. Since his return to Québec City at Defence R&D Canada – Valcartier (DRDC Valcartier), he has been responsible for research in command and control: system architectures, concurrent computing, wide-area naval picture (1997, co-author of an AUS-CAN-NZ-UK-US document on naval information management used in NATO concept of operations for information management and performance measures; recipient of a best paper award in 1998 related to this work). Elected to the rank of IEEE Senior Member in 1997. He graduated, BASc in 1978 and M.Sc. in 1980, from Laval University in Electrical Engineering. *Address for correspondence:* System of Systems Section, Defence Research Establishment Valcartier, 2459 Pie-XI Blvd. North Val-Bélair, (Québec), Canada, G3J 1X5, E-mail: paul.labbe@drev.dnd.ca; Defence R&D Canada - Valcartier, Val-Bé lair QC G3J 1X5, Canada.

**BACK TO TOP**

---

# A Software Agent-based Collaborative Approach for Humanitarian-Assistance Scenarios

*Zakaria Maamar, Nabil Sahli, Bernard Moulin, and Paul Labbé*

**Keywords:** Software agent, collaboration, humanitarian operations, humanitarian-assistance scenarios

**Abstract:** This paper presents a research project that deals with the use of software agents as a support to collaboration. Collaboration can face different obstacles, such as partner distribution and resource heterogeneity. To overcome these constraints, coordination strategies are required. Such strategies allow agents to avoid conflicts and, hence, to fulfill their activities efficiently. This project is applied to humanitarian-assistance scenarios, in which different participants, such as non-governmental organizations, have to work together despite their individual differences. Therefore, it becomes relevant to suggest collaborative approaches that will support these participants in their daily operations.

**full text**

Authors: **Zakaria Maamar and Paul Labbé**
Title: **Multi-Agent Systems to Support Coalition Forces**
Year of issuance: **2002**
Issue: **Information & Security. Volume 8, Number 2, 2002, pages 156-174**
Hard copy: **ISSN 1311-1493**

---

# MULTI-AGENT SYSTEMS TO SUPPORT COALITION FORCES

Zakaria MAAMAR and Paul LABBÉ

---

**Table Of Contents:**



---

## 1. Introduction

We discuss our work on the design and development of collaborative environments for distributed and heterogeneous military applications. These applications, called Command and 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 and rescue, and emergency services. In a military context, a commander makes decisions concerning his troops deployment using the information that the CCIS supplies. It may occur that the commander aims at involving other friendly CCISs before making his decisions. For example, a commander of Nation X has to take into account the positions of the enemy and friendly troops. Therefore, he has to involve other CCISs that may possess such information. It would be more appropriate if this commander could perform this operation without being aware of the characteristics of each CCIS. 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.*[1]):

- *Autonomy:* in the coalition environment, CCISs should have the flexibility to be designed, developed, and managed independently, without having to comply with the standards of this environment.
- *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 the 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 coalitions.

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 can be viewed as a collection of collaborative MASs; each MAS corresponds to a CCIS and each MAS contains different types of agents, fulfilling different roles, and achieving different responsibilities.

In the IC2MAS interoperability approach, MASs are the front-ends of the CCISs to the coalition network and hence, have the capability to act on their behalf. Moreover, MASs encompass different Software Agents (SAs) [2] that handle and perform the functionality 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 where the network infrastructure is not reliable), some of the SAs in the IC2MAS approach are able to create Slave-Agents [3] and enhance them with mobility mechanisms.[4] 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 forces of Nation X have to interact with non-government organizations as well as with armed forces of other countries. Therefore, SAs have to rely on communication [5] and coordination [6] mechanisms to avoid conflicts and collaborate efficiently. When diverse SAs communicate, they have to understand each other. By establishing ontology,[7] 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) 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. Degrees of Interoperability in a Coalition*

In a coalition, three degrees of interoperability are identified (adapted from Au *et.al*).[8] Basic interoperability, called inter-connectivity, 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, collaboration at the commandment level is needed. In what follows, the three degrees of interoperability are summarized as follows (cf. Figure 1).

- *Physical inter-connectivity:* to guarantee basic communication, computing resources are first interconnected to exchange messages. This inter-connectivity 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 they are heterogeneous at different levels (hardware, software, and terminology).
- *Commandment collaboration:* it goes beyond the integration of applications, by expanding military operational processes to other structures. To this end, a collection of components, such as software agents gathered into multi-agent systems, could be set up. These components collaborate more than just inter-operate.

Figure 1, the commandment level relies on the application level to achieve the coalition's main-mission, for example humanitarian assistance. Furthermore, the commandment level interacts regularly with its headquarters. The purpose is to keep the headquarters 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, weather forecast, and logistics. 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.

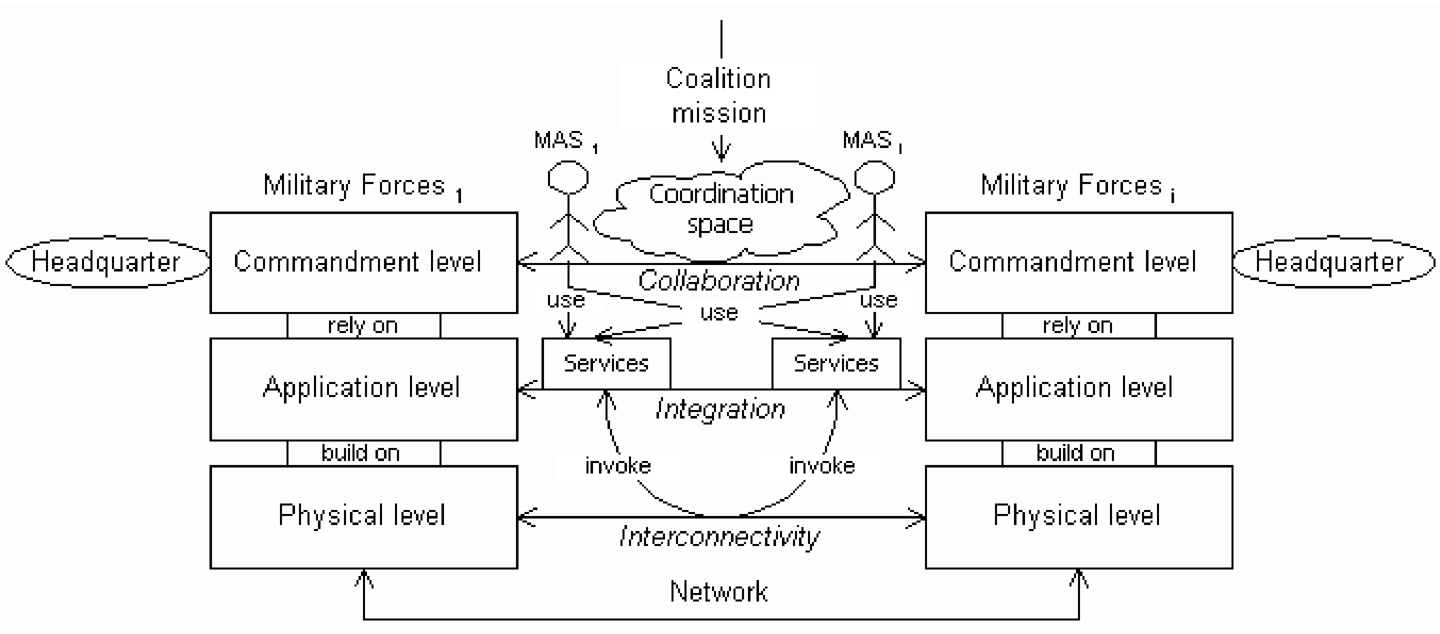



**Figure 1:** From inter-connectivity to collaboration, through integration

### *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.* a CCIS consists of a structure, functions, and tasks.[9] The CCIS structure represents an assembly of facilities, arranged to meet the objectives of the CCIS. To reach these objectives, the functions of the CCIS are initiated in order to carry out the needed tasks. Tasks require the structure's facilities, in terms of personnel, 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.

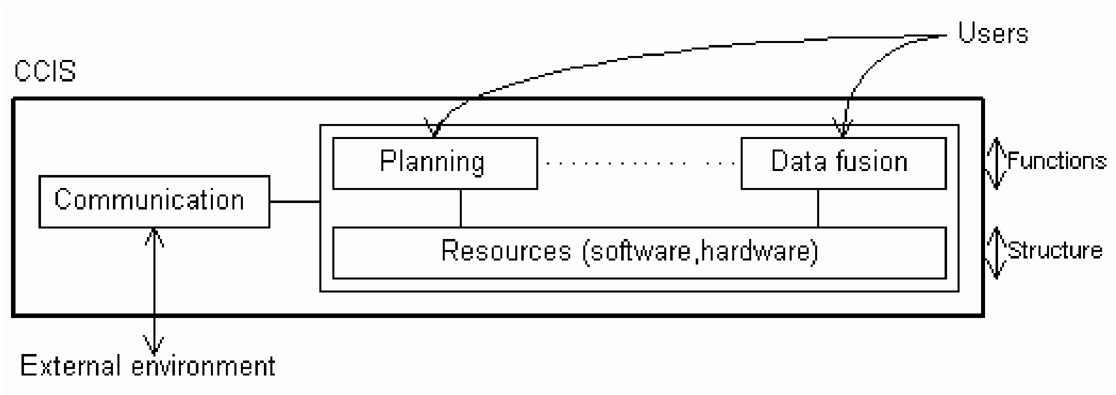



Figure 2: CCIS Simplified Architecture

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 mode is detailed.

### *3.1. Requirements of a Coalition*

In the IC2MAS project, the running scenario corresponds to a coalition that is set up by different countries for diverse purposes, e.g. 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 better equipped and prepared than others.
- At diverse hierarchical levels, different people make decisions during the course of operations. It happens that a decision is based on 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, the task to provide and maintain a high level of assistance to military users is very complex. 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 can be exchanged, what types of operations can be delegated, and what types of communication approaches can be used. There is a research avenue associated with each requirement.

- Requirement: What types of information can be exchanged?
  Research Avenue: Ontology.
  Definition: Ontology is a means to express and exchange information that is understood by all participants in a coalition. Moreover, to be used efficiently, ontology requires a language for representation (e.g. KIF) and a language for communication (e.g. KQML).
- Requirement: What types of operations can 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 by Labrou *et.al.*,[10] communities of agents are much powerful than any individual agent.
- Requirement: What communication approaches can 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 be moved to a common workplace.

### *3.2. IC2MAS Architecture*

In the literature, different approaches that deal with the problem of inter-operable systems can be found, among them Infosleuth,[11] TSIMMIS,[12] SIMS,[13] and SIGAL.[14] All these approaches agree on the use of SAs as a means to develop such systems, and have several characteristics in common. The SAs they use are static. Therefore, these SAs do not have the ability to move to distant systems. Moreover, all these approaches assume that the network infrastructure is fully reliable and has unlimited bandwidth for information transmission.

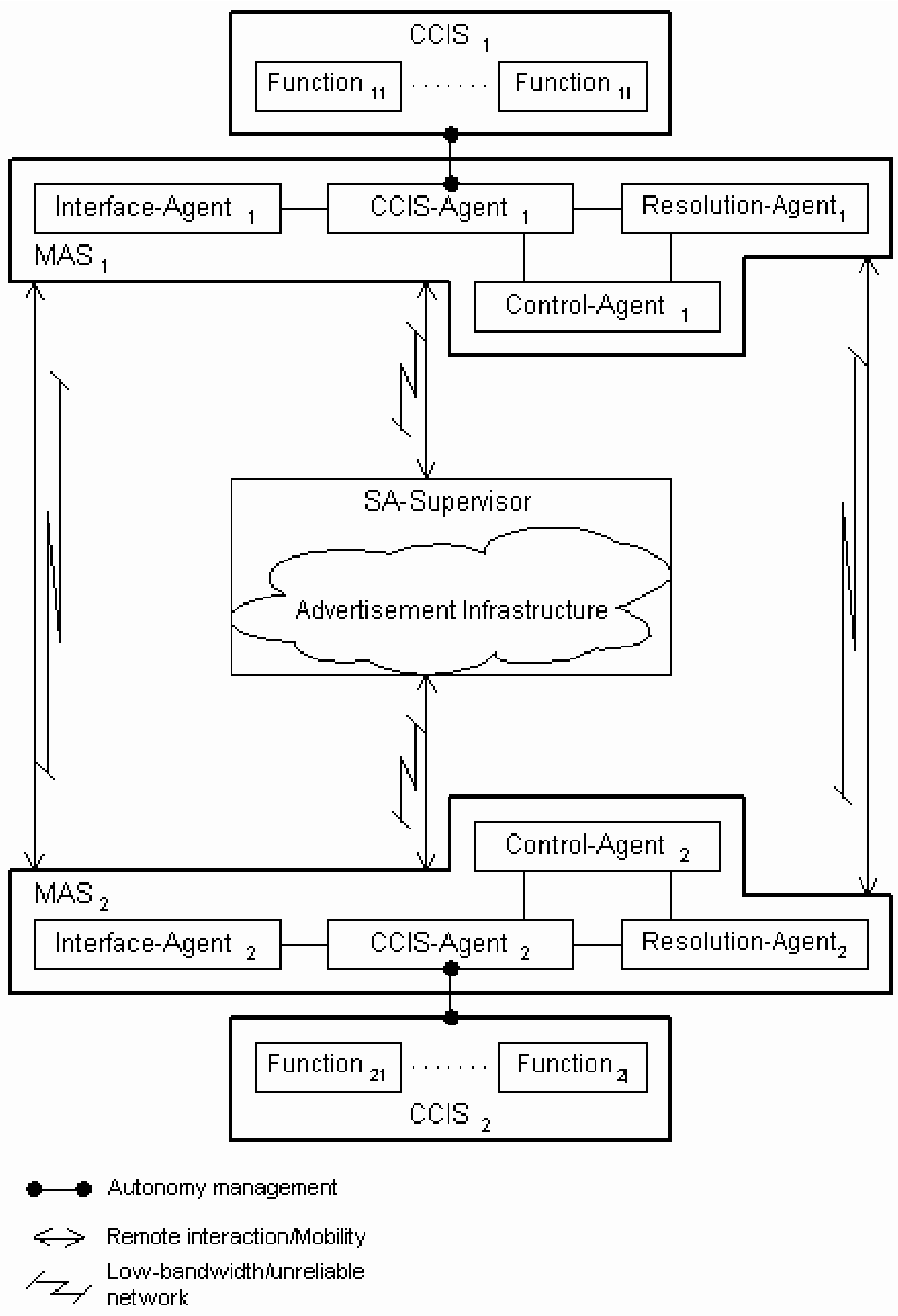



**Figure 3:** IC2MAS Architecture for coalition support

Based on these different approaches and considering the requirements of a coalition, we have proposed an IC2MAS architecture for application in a coalition context (cf. Figure 3). Multiple MASs form the backbone of this architecture and interact both remotely and 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.

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. The creation process of Slave-Agents complies with the Supervisor-Worker pattern as defined by Fischmeister and Lugmayr.[15] In the next sections, agents' functionality is depicted.

### *3.3. Software Agents and the Advertisement Infrastructure*

There are various types of SAs within the IC2MAS architecture. These SAs belong to different MASs. They collaborate through the Advertisement-Infrastructure facility. Details of the internal-modules of the agents are provided below.

**Interface-Agent.** By analogy to Interface-Agents of Maamar *et.al.*[16] and Sycara *et.al.*,[17] the IC2MAS 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.

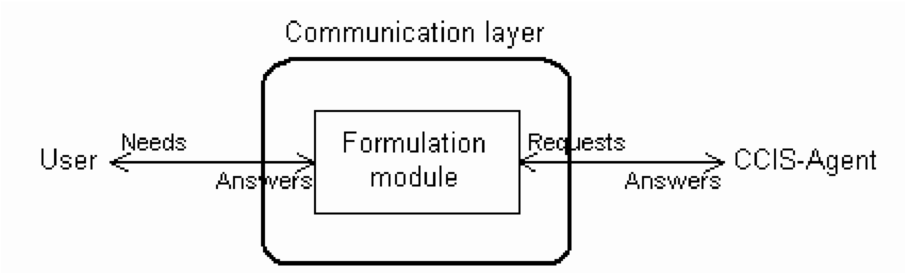



**Figure 4:** Interface-Agent modules

The Interface-Agent consists of one module, called *Formulation module* 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.*[18] and Task-Agents of Sycara *et.al.*,[19] 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.*, [20] the IC2MAS' CCIS-Agent acts on behalf of CCIS and, hence, maintains its autonomy towards the coalition. To achieve this autonomy, the CCIS-Agent advertises, through its services, the functions its CCIS performs.[21] 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.

A CCIS offers different functions that vary from data fusion and weather forecast to planning (cf. Section 2.2). Considering these functions and the complexity of CCISs (for instance, a planning function could be a distributed-object client/server application running on top of an Object Request Broker middleware) a new type of SAs, called *Function-Agents*, is introduced in the IC2MAS architecture at the MAS level. Each Function-Agent is associated with a CCIS 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 results back. IC2MAS Function-Agents are similar to Information-Agents described by Sycara *et.al.*[22]

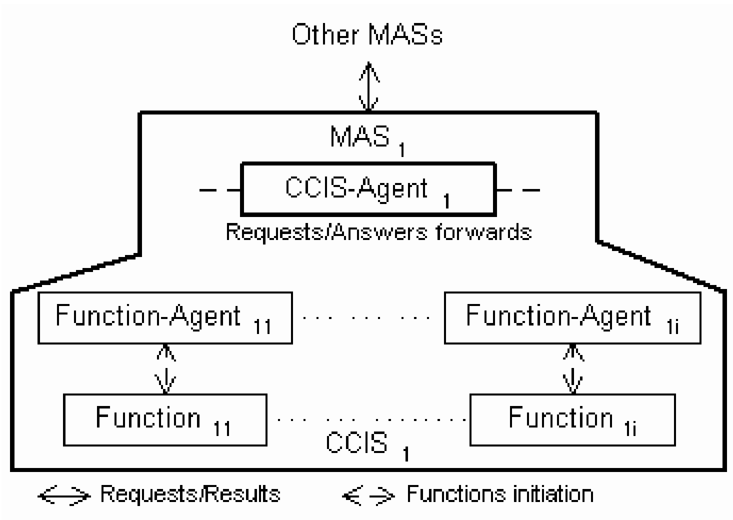



**Figure 5:** Function-Agents at the MAS level

Figure 6 presents the modules of CCIS-Agents and Function-Agents. Similarly to the Interface-Agent, a communication layer encapsulates the two agent 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 processes 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 reported to the CCIS-Agent's definition-module.

**Resolution-Agent.** By analogy to Resolution-Agents of Maamar *et.al.* [23] and Task-Agents of Sycara *et.al.*,[24] the IC2MAS's Resolution-Agent processes users' requests, only if these requests have been 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 search for in the Bulletin Board.

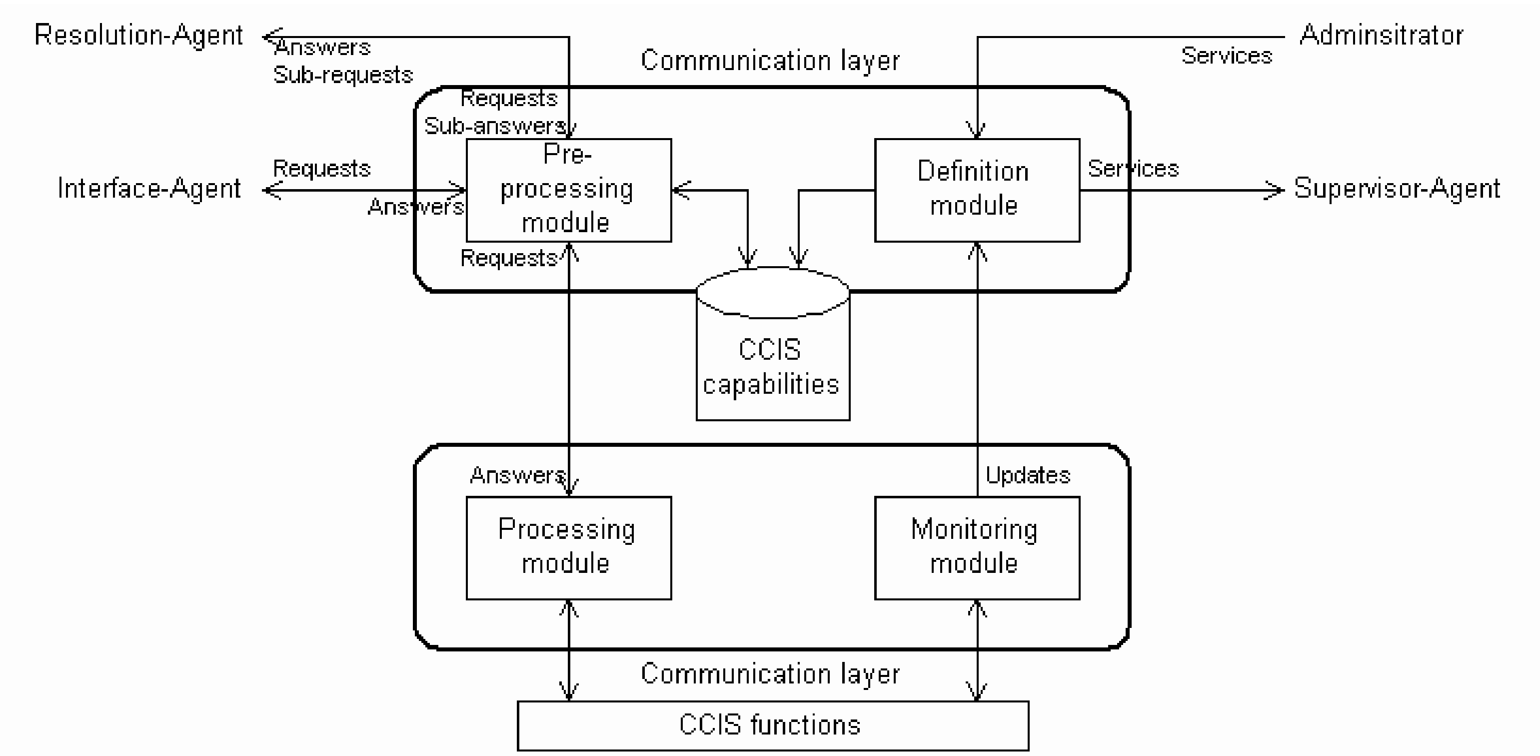



**Figure 6:** CCIS-Agent and Function-Agent modules

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 that is needed in order to perform the user's request. Generally, such a procedure is called a route. Then, the Resolution-Agent creates another Slave-Agent, called Route-Agent, and assigns to it this procedure. The Route-Agent may either interact remotely with the CCIS-Agents of other MASs or migrate to different MASs and meet locally with their CCIS-Agents. A decision about a remote request or mobility is based on the network status and the number of the CCISs required in satisfying users' requests. Similarly to CCIS-Agents, a security level is also associated with Slave-Agents. This security level is used to check Slave-Agents entering the Advertisement-Infrastructure and the various 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.

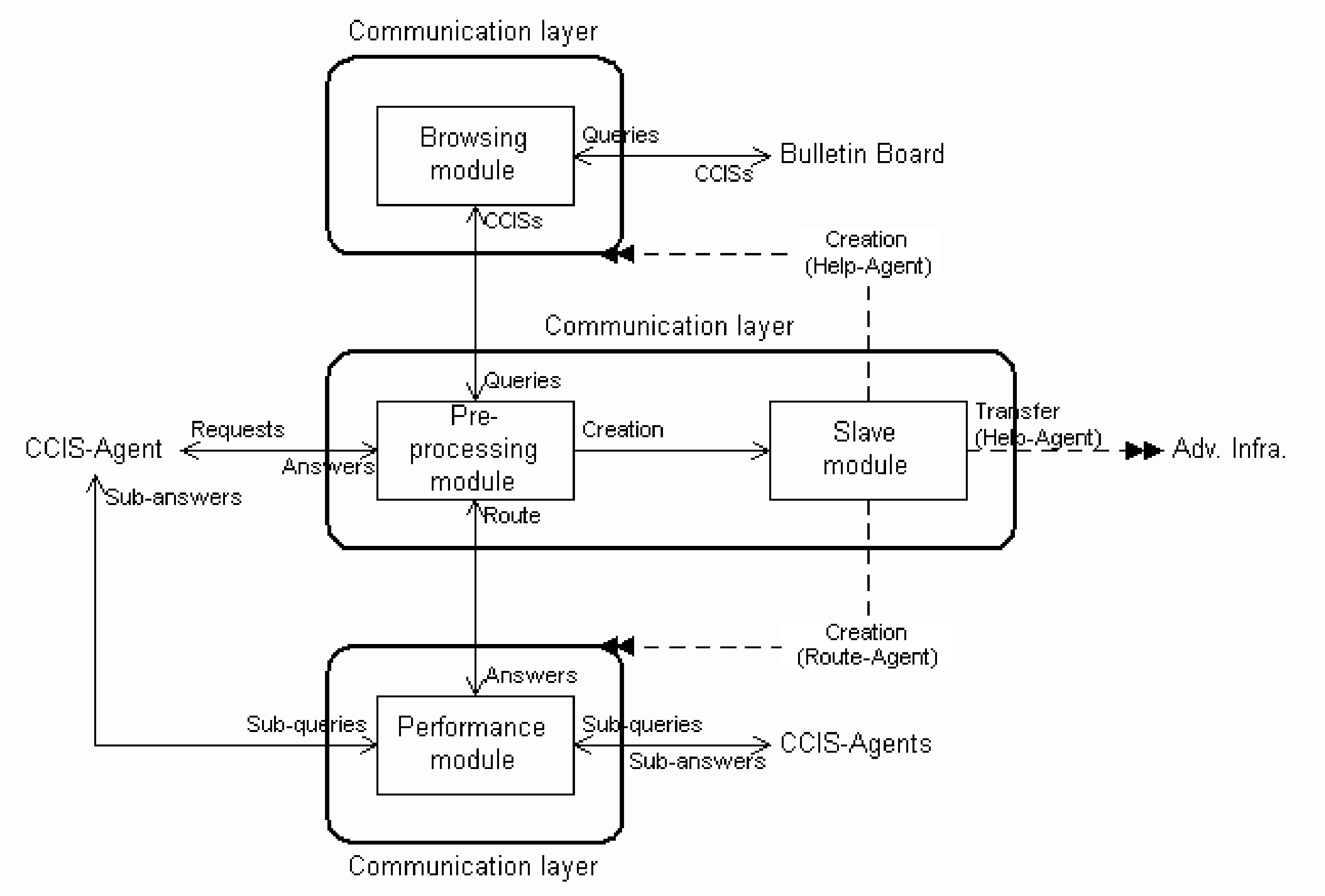



**Figure 7:** Resolution-Agent modules (including Help-Agent and Route-Agent)

**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 the advertisements of CCIS-Agents, 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 updated also 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 Radio to send and request information. The military users often rely 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 in operating mode*

Based on the characteristics of the IC2MAS architecture and the types of SAs this architecture integrates, we proposed four stages to handle the operation of IC2MAS: Initialization, Advertisement, Operation, and Maintenance. This section describes the features of each stage. Note that Initialization and Advertisement stages are transparent to users.

#### *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 establishes the security policy that manages the access of agents 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 8). As soon as it arrives, the Help-Agent is checked, registered, and finally installed.

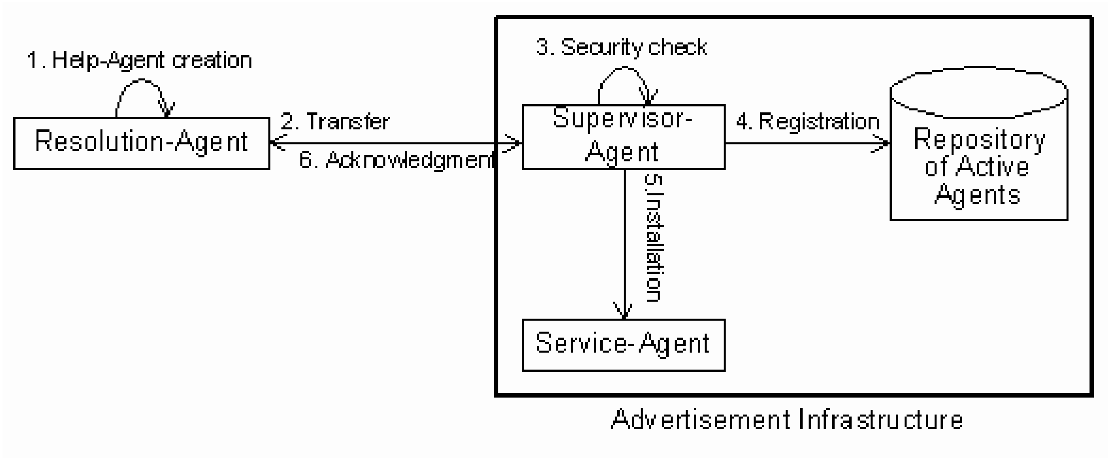



**Figure 8:** Help-Agent in the Advertisement Infrastructure

For the case considered we assume that before leaving and entering MASs, Help-Agents and Route-Agents interact with Control-Agents for security, shipping, and installation purposes.

***Advertisement Stage***

Once initialization is complete, 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 9).

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 will be offered. 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 evaluates and acknowledges the CCIS-Agent in case the operation is successful or makes the CCIS-Agent aware of the reasons for failure. 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 to either update or withdraw the advertised services.

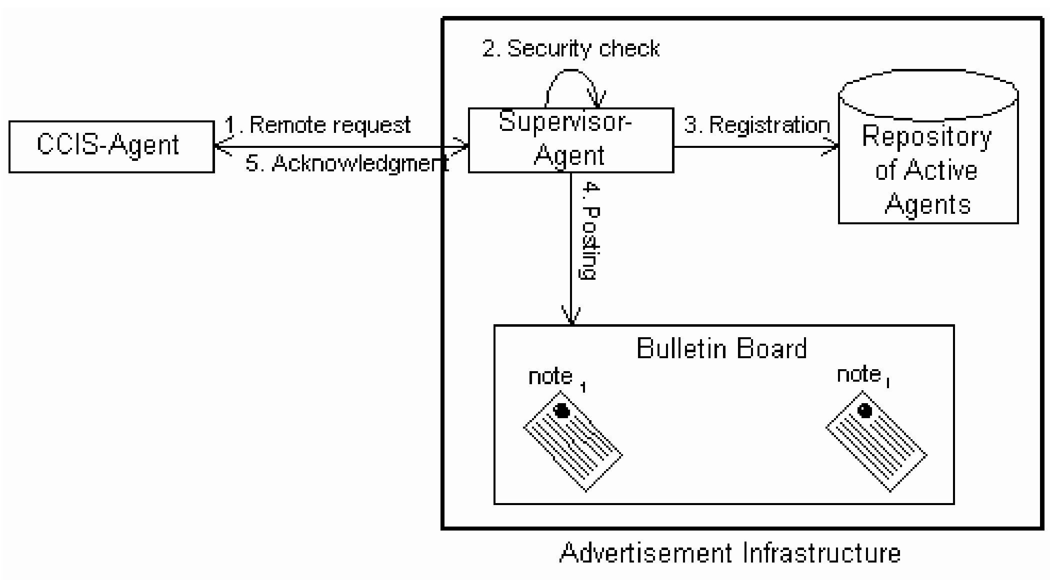



**Figure 9:** Services advertisement in the Bulletin Board

*Operation Stage*

Once the advertisement stage is accomplished, the IC2MAS environment is ready for operation. The operation stage of IC2MAS is summarized by two situations (cf. Figure 10):

- Only the user's CCIS is required: the CCIS-Agent is in charge of handling this situation (cf. Figure 10-a).
- Several CCISs, including or not the user's CCIS, are required: the Resolution-Agent is in charge of handling this situation (cf. Figure 10-b).

In what follows, numbers in parenthesis illustrate the chronology of operations in Figure 10.

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 10 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 contacts and interacts remotely with its Help-Agent in order to identify appropriate CCISs (4.b). Once the Help-Agent has completed its operation (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-A gent 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).

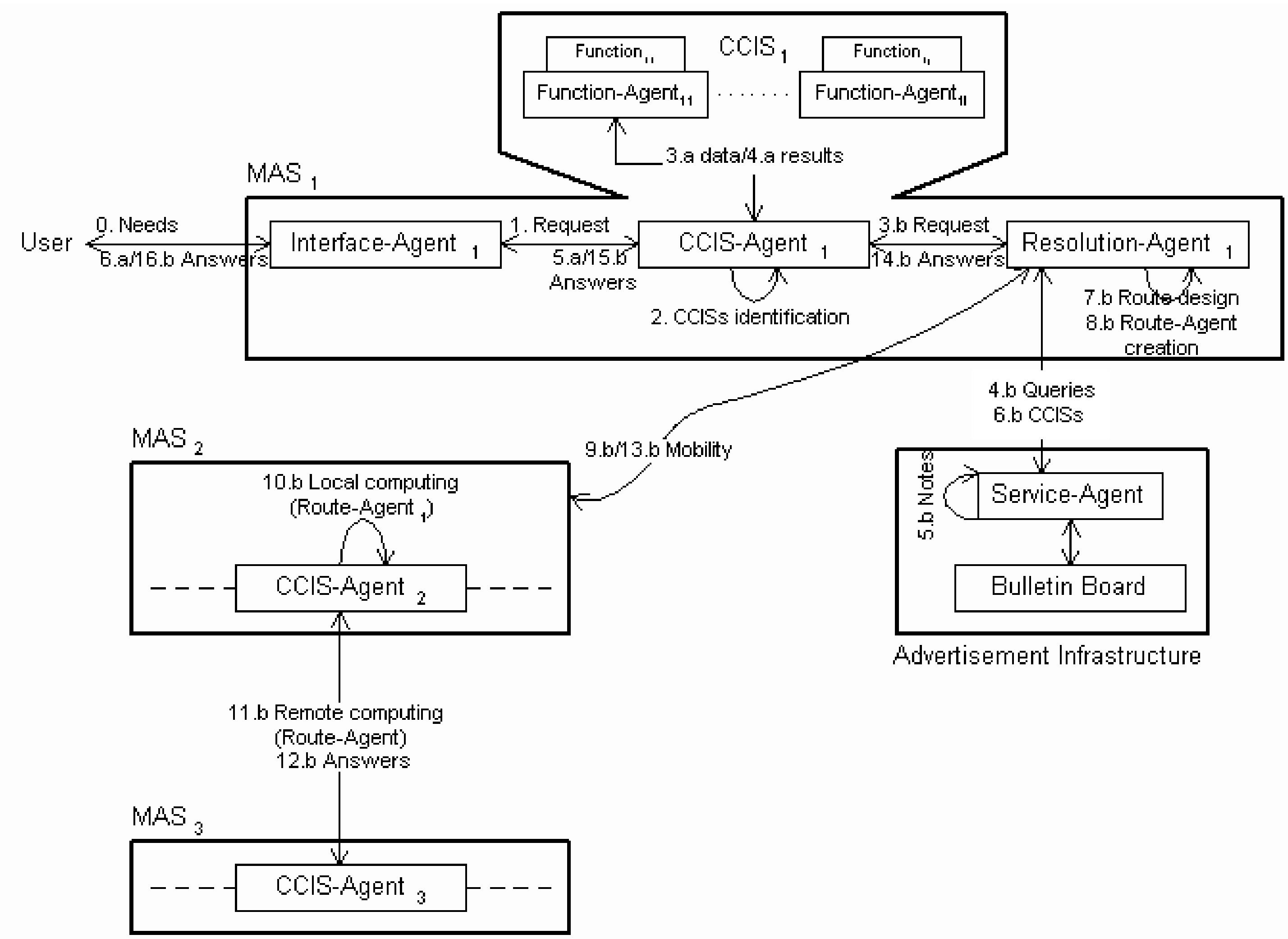


**Figure 10:** User’s request satisfaction

To clarify things, hereafter is an example illustrating this itinerary. In Figure 10, the itinerary indicates that the Route-Agent first has to move to $MAS_2$ (9.b). Next, the Route-Agent interacts locally with the CCIS-Agent of this $MAS_2$ (10.b). Furthermore, to complete its operations, the itinerary shows that the Route-Agent has to interact remotely with CCIS-$Agent_3$ of $MAS_3$. Then, the Route-Agent sends a request (11.b) and waits for the results from CCIS-$Agent_3$ (12.b). Next, the Route-Agent goes back to its original MAS (13.b) and interacts with Resolution-Agent’s 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).

#### *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 and operation of the IC2MAS environment. Several situations have been identified. In this paper, we briefly present two of them:

- It happens that a CCIS adapts its structural and functional characteristics, for example by adding a new function. Therefore, the CCIS-Agent has to be adapted by adding new services to its capabilities
- It happens that the Supervisor-Agent cleans up the Bulletin Board of the Advertisement Infrastructure, because of 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. Different research projects exist in the field of systems interoperability.[25,26] All these projects have the same concerns, namely:

- To 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.
- To reduce the informational disparities between the integrated systems. In the IC2MAS environment, the definition of ontology is, currently, tackled (cf. Section 5.1).
- To 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 taken into account in the IC2MAS environment. For example, a security policy manages the Advertisement Infrastructure and a security level is used to identify agents. Additional security elements could be suggested, for instance identifying services with authorization levels and users with use levels.

### 5. Current Efforts in ICMAS

This section gives insights on topics that are currently tackled in the IC2MAS environment. First, the ontological disparities are briefly described. Ontology is one of the main issues to be addressed in the design of an interoperable environment for heterogeneous systems. We consider ontology as a means to represent and exchange information that is 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 the capabilities of his CCIS (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 individual differences, for example diversity of cultures that exist among the participants in a coalition. 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 the level of users.

## 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 was considered. 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, and Route-Agent) while four stages describe this architecture in operating mode (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 a capability of providing a "fully" automated coalition support. A whole set of negotiations, dialogues, coordination and communication among participants, groups of participants, and systems are involved.

---

**Notes:**


1. G. Babin, C. Hsu, and Z. Maamar, *A distributed metadatabase architecture for an enterprise scalable, adaptive integration environment*, Technical Report 37-94-424 (Troy, N.Y., USA: DSES, Rensselaer Polytechnic Institute, 1994).

2. S. Green, L. Hurst, B. Nangle, P. Cunningham, F. Somers, and R. Evans, *Software agents: A review*, Technical report (Dublin, Trinity College and Broadcom Eireann Research Ltd., May 1997).

3. F. Buschmann, R. Meunier, H. Rohnet, P. Sommerland, and M. Stal, *Pattern-Oriented Software Architecture* (John Wiley and Sons Ltd., 1996).

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

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

6. H. Ghenniwa, *Coordination in Cooperative Distributed Systems*, PhD thesis (Ontario, Canada: University of Waterloo, Systems Design Engineering Department, 1997).
   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).

7. D. Jones, T. Bench-Capon, and P. Visser, "Methodologies for ontology development," in *Proceedings of IT&KNOWS - Information Technology and Knowledge Systems, 15th IFIP World Computer Congress* (Vienna, Austria, and Budapest, Hungary, 1998).

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

9. S. Malerud, E.H. Feet, and U. Thorsen, *A method for analysing command and control systems,* (FFI) N-220 (Kjeller, Norway: Norwegian Defence Research Establishment).

10. Labrou, Finin, and Peng, "Agent communication languages: The current landscape."

11. 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 *Proceedings of the 1997 ACM International Conference on the Management of Data SIGMOD* (Tucson, Arizona, 1997).

12. 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 *Proceedings of the 10th IPSJ* (Tokyo, Japan, 1994).

13. C.A. Knoblock, Y. Arens, and C.N. Hsu, "Cooperating agents for information retrieval," in *Proceedings of the Second International Conference on Cooperative Information Systems* (Toronto, Canada, 1994).

14. Z. Maamar, B. Moulin, and Y. B•dard, "Software agent-oriented frameworks for the interoperability of georeferenced digital libraries on the World Wide Web: The SIGAL project," in ed. R. Fegeas M.F. Goodchild, M.J. Egenhofer and C.A. Kottman, *Interoperating Geographic Information Systems* (Boston: Kluwer Academic Publishers, 1999): 335-354.

15. S. Fischmeister and W. Lugmayr, “The supervisor-worker pattern,” in *Proceedings of Pattern Languages of Programs PLoP'99* (Kubova Hut, Czech Republic, 1999).

16. Maamar, Moulin, and B•dard, “Software agent-oriented frameworks for the interoperability.”

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

18. Maamar, Moulin, and B•dard, “Software agent-oriented frameworks for the interoperability.”

19. Sycara, Decker, Pannu, Williamson, and Zeng, “Distributed intelligent agents.”

20. Maamar, Moulin, and B•dard, “Software agent-oriented frameworks for the interoperability.”

21. Currently, the services do not have constraints, e.g. cost.

22. Sycara, Decker, Pannu, Williamson, and Zeng, “Distributed intelligent agents.”

23. Maamar, Moulin, and B•dard, “Software agent-oriented frameworks for the interoperability.”

24. Sycara, Decker, Pannu, Williamson, and Zeng, “Distributed intelligent agents.”

25. R. Bayardo et al., “Infosleuth: Semantic integration of information in open and dynamic environments”; S. Chawathe et al., “The TSIMMIS project: Integration of heterogeneous information sources.”

26. M.R. Genesereth and A.P. Ketchpel, “Software agents,” *Communication of the ACM* 37, 7 (July 1994):48-53; C. Hsu, *Enterprise Integration and Modeling - the Metadatabase Approach* (Boston, Mass., USA: Kluwer Academic Publisher, 1996); M.P. Papazoglou, S.C. Laufmann and T.K. Sellis, “An organizational framework for cooperating intelligent information systems,” *International Journal of Intelligent and Cooperative Information Systems* 1, 1 (1992): 169-202.

---

**ZAKARIA MAAMAR** is an associate professor at the College of Information Systems of Zayed University. In addition, he is respectively an adjunct professor at the Computer Sciences Department and an associate researcher at the Research Center in Geomatics, both of Laval University, Quebec, Canada. Dr. Maamar received two IEEE awards in 1999 and 2001, respectively. Dr. Maamar’s research interests are mainly related to software agents, wireless systems, and web and mobile services. Dr. Maamar has published several papers in journals and conference proceedings. Dr. Maamar graduated for his M.Sc. and Ph.D. in computer sciences from Laval University, Quebec, Canada in 1995 and 1998, respectively. *Address for correspondence:* Software Agents Research Group, College of Information Systems, Zayed University, PO Box 19282, Dubai, United Arab Emirates, E-mail: zakaria.maamar@zu.ac.ae; Research Center in Geomatics and Computer Science Department, Laval University, Ste-Foy QC G1K 7P4, Canada.

**PAUL LABBÉ** joined the National Defense Canada in 1978. While at the Communications Research Centre in Ottawa 1984-89, he was responsible for the research activities on naval communications architectures and data network. Since his return to Québec City at Defence R&D Canada – Valcartier (DRDC Valcartier), he has been responsible for research in command and control: system architectures, concurrent computing, wide-area naval picture (1997, co-author of an AUS-CAN-NZ-UK-US document on naval information management used in NATO concept of operations for information management and performance measures; recipient of a best paper award in 1998 related to this work). Elected to the rank of IEEE Senior Member in 1997. He graduated, BASc in 1978 and M.Sc. in 1980, from Laval University in Electrical Engineering. *Address for correspondence:* System of Systems Section, Defence Research Establishment Valcartier, 2459 Pie-XI Blvd. North Val-Bélair, (Québec), Canada, G3J 1X5, E-mail: paul.labbe@drev.dnd.ca; Defence R&D Canada - Valcartier, Val-Bé lair QC G3J 1X5, Canada.

**BACK TO TOP**

---


**Information & Security. An International Journal**
**e-mail: infosec@mbox.digsys.bg**

# Multi-Agent Systems to Support Coalition Forces

*Zakaria Maamar and Paul Labbé*

**Keywords:** Software agents, coalition, Command and Control Information Systems, CCIS

**Abstract:** The distributed, heterogeneous, 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) gathered into MultiAgent Systems (MASs) seem to be a promising approach. To develop this approach, different aspects of a coalition have 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.

**full text**

Author: **Marvin L. "Lenard" Simpson, Jr.**
Title: **Integration and Interoperability of Information Systems within the Coalition Aerospace Operations Center**
Year of issuance: **2002**
Issue: **Information & Security. Volume 8, Number 2, 2002, pages 175-193**
Hard copy: **ISSN 1311-1493**

---

# INTEGRATION AND INTEROPERABILITY OF INFORMATION SYSTEMS WITHIN THE COALITION AEROSPACE OPERATIONS CENTER

Marvin L. "Lenard" SIMPSON, Jr.

---

**Table Of Contents:**



---

## Introduction

In any coalition operation we face a continuing challenge where we must strike a balance between classified information sharing among coalition partners and a requirement to protect each coalition partner's information sources and collection capabilities as defined by each nation's laws and regulations.

With that in mind, in order to maximize combat capability and reduce risk (fratricide, threat avoidance and suppression, inadvertent disclosure of sensitive data etc.) the operational commander and his staff must have access to the most accurate information in time to plan, act and react with confidence.

When information/data is stored, sorted and manipulated across several different Local Area Networks (LAN's) and Wide Area Networks (WAN's), each with disparate security levels and applications, we run the risk that critical information will not be available for use by the appropriate personnel in a timely enough manner to make the correct decision.

Unfortunately, current technology does not support automatic transfer and synchronization of massive databases across LAN's and WAN's with disparate security requirements. With that in mind, perhaps the answer is one master data set populated with the best information available and usable by all coalition partners. This master data set would be used to plan and execute the majority of coalition military activities. There is an inherent risk associated with this arrangement, which will be covered in this paper.

## CAOC Defined

A Coalition Aerospace Operations Center (CAOC) is defined for this paper as the location/organization (personnel, capabilities and equipment) through which the Coalition Forces Air Component Commander (CFACC) exercises command and control ($C^2$) of aerospace forces. It is the senior element of the Theater Air Control System (TACS).

The CFACC employs the CAOC to facilitate the maneuver and mass overwhelming aerospace power through centralized control and decentralized execution to produce desired operational and strategic effects in support of the Coalition's campaign. Infrastructure, systems, processes, and training should be shared and integrated to the maximum extent possible while ensuring that the integrity of all military missions is maintained. The design should maximize interoperability while promoting the independence and flexibility necessary to support complementary—but not identical—missions executed under nominal conditions.

### *Assumptions*

The CAOC functions at the coalition/component level and provides the CFACC with the capability to direct and supervise the activities of assigned, supporting, or attached forces, and to monitor the actions of both enemy and friendly forces. In order to function, the CAOC requires connectivity to operations centers of higher service/joint/coalition headquarters, lateral headquarters, and subordinate units. This allows for the continuous collection and presentation of battle management information.

CAOC personnel, in accordance with the priorities, objectives, and strategies, conduct detailed direction of all aerospace operations by using this data.

To lay out a proposed methodology for a notional CAOC, several assumptions have to be made. It will be assumed that the Coalition commander exercises command over the Coalition command, control, communications, and computers intelligence ($C^4I$) information and processes system. The Coalition will use its $C^4I$ System to plan, direct, coordinate, and control the various aspects of Coalition operations. These missions could include but are not limited to:

1. Maintain continuous surveillance of the Area of Responsibility (AOR) aerospace to deter hostile states from entering the AOR.
2. Effectively employing assigned forces in defense of the AOR should deterrence fail.
3. Planning, directing, monitoring, and controlling air operations while providing $C^2$ support to assigned forces and other military and civilian agencies.
4. Integrating their operations with other $C^2$ systems to form a coherent structure for joint and combined operations.

### *Processes*

The primary processes used in our notional CAOC will be the same as used by current AOC's, which are:

#### *1. Planning*

Aerospace planning processes will focus on the desired strategic and operational effects the CFACC is to produce. These desired effects will be articulated in courses of action (COAs). Once a COA is selected, the effects the CFACC is tasked to produce will be specified. The aerospace strategy is the CFACC's concept for employing aerospace capabilities to achieve objectives in support of the overall campaign. The "means" will be kinetic and/or non-kinetic use of aerospace power. Since aerospace power is a theater-wide instrument, CFACC planners must be involved in the development of other coalition member COA options. This integration of plans constitutes three, highly iterative and interactive, sub-processes. One is developing support requirements those components foresee from the aerospace component. The other is the support requirement that the aerospace component foresees from other components. A third— and perhaps the most important— sub-process is gaining an understanding of the mechanism each component foresees on how their COA helps accomplish the overall battle objectives. Integration of efforts between components occurs on multiple levels requesting aerospace support. The focus here is integration of the CFACC's planners through the execution and assessment process to maintain a long-term focus concurrently with the other components/ coalition member country.

Based upon changes in the situation, direction or resources, planners will develop daily CFACC guidance for approval and dissemination. Guidance, translated into prioritized aerospace tasks, provides the necessary information to begin the target nomination process. The targeting process is the linkage between guidance and application. Targets are nominated based on this guidance, intelligence recommendations, component requests, and other factors. Resources are then allocated to accomplish specific missions. Packages are constructed to maximize the effectiveness of available assets. The Master Air Attack Planning cell (MAAP) provides specific guidance on how daily aerospace operations will be conducted. The MAAP provides theater-level sequencing and resource inputs necessary for producing an ATO and is the first time in the aerospace tasking process that detailed resource availability is matched against specific targets to achieve desired effects. Working closely with specialty teams, component liaisons, and unit representatives, the Integrated Prioritized Target List, threat situation, joint prioritized collection, forecast weather, weapons system availability, air refueling, and weapons employment options are synchronized. The MAAP has sufficient flexibility to adapt to the changing battlefield situation throughout the theater.

After the MAAP's plan is approved by the CFACC, detailed preparations continue with the production of the ATO, Special Instructions (SPINS), and Airspace Control Order (ACO) using CFACC guidance, target worksheets, and various component inputs. The Airspace Control Authority and Area Air Defense Commander instructions provide sufficient detail to allow components to plan and execute all missions tasked. These directions enable combat operations without undue restrictions, balancing combat effectiveness with the safe, orderly, and expeditious use of airspace.

### *2. Execution*

Even a perfect plan requires adjustment during the execution phase of conflict. Execution comprises steady state and dynamic functions enabling the CFACC command of continuous, rapid, and dynamic aerospace power. The CFACC commands aerospace power across the spectrum of conflict by directing, controlling, monitoring, and assessing forces under his tactical control. The CFACC will maintain battlespace awareness, which is essentially situational awareness at the operational level. Battlespace awareness will provide an accurate picture of friendly and enemy operations within the area of interest and is the key enabler of the CFACC's ability to command in real time. It also provides the capability to view and monitor emergent threats and potential targets giving the CFACC the ability to redirect assets during execution. Key data and information links will be from, as a minimum, in-theater and national level sensors, tactical data links, and messaging at all levels to provide relevant information available for execution. Battlespace Awareness, in a Common Operational Picture (COP) for example, combines information from air, surface, subsurface, ground, and space assets to provide a three-dimensional view of the battlespace. Sensor and data fusion within this picture plays an important role in validating targets and eliminating ambiguous information.

### *3. Continuous Assessment*

The CAOC will monitor enemy and friendly actions and reactions; identify potential threats, weather and its impact to friendly forces/operations, and subordinate unit combat reports/logistics status. During execution, each functional area will continuously monitor changes to the plan. This information provides a comprehensive picture of current and projected capability, and is tailored to the needs of the CFACC. In addition to speeding up the decision making process, this "quick look" picture facilitates proactive command of aerospace operations. Developing this level of situational awareness should not inundate decision makers, but rather give them information needed to command/control the fight, and answer the CFACC question, “What’s the enemy doing and what options/capability do we have at this point?” As unexpected/unplanned events occur which affect the plan (e.g. strike package delays), combat operations will assess impact on their own functional area, the impact on the current operation, level of reporting required, and then develop options for the decision maker. Changes must be communicated horizontally to other planners, and vertically to tactical units and other headquarters. Continuous assessment often reveals a “trigger event,” that can launch combat process.

## Current Information Protocols

### *Foreign Disclosure Officers*

Foreign Disclosure officers are the personnel in the US CAOC responsible for implementing the National Disclosure Policy (NDP) that governs the disclosure of United States Classified Military Information (CMI) to foreign governments and international organizations. CMI is under the control of the US Department of Defense. Access to CMI is based on the impact to national security. It is designated as TOP SECRET, SECRET or CONFIDENTIAL. Foreign Disclosure officer responsibilities include: Knowledge of specific disclosure criteria and limitations, definitions of terms, and other guidance governing decisions on the disclosure of CMI.

Under conditions of actual or imminent hostilities, any Unified Commander may disclose CMI through TOP SECRET to an actively participating allied force when support of combined combat operations requires it. The U.S. Commander must notify the Chairman of the Joint Chiefs of Staff (JCS) of such disclosures. The Chairman of the JCS, in turn, must notify the Office of the Under Secretary of Defense for Policy, who will determine whether any limitations are necessary on continued disclosure of the information.

### *Current Automatic Assurance Guards*

Current automatic information assurance guards are similar to the Defense Intelligence Agency - certified Imagery Support Server Environment (ISSE) Guard, developed by Rome Laboratory. The ISSE Guard provides a secure interface for the direct soft-copy exchange of information between Top Secret Special Compartmental Information (SCI) systems and Secret Collateral systems operating over strategic and tactical wide or local area networks. The Guard consists of the Common Guard Interface (CGI) application, hosted on high side users’ workstations and the Guard application running on the B-1 certified CyberGuard Night Hawk platform. The Guard is a bidirectional guard supporting the high to low and low to high transfer of e-mail and image files and the high to low transfer of text files. The ISSE Guard currently has two external interfaces: a high side Ethernet (IEEE 802.3) interface and low side interface that can be either 802.3 or X.25. ISSE Guard provides the functionality required to securely connect, validate, downgrade and transfer information between systems and networks operating at different security levels, while the CGI provides high side users with an interface to the Guard.

Imagery Support Server Environment Guard version 3.0 permits the secure digital exchange of electronic mail, imagery, text, and multimedia information between networks operating at dissimilar security classification levels. The system provides the ability to electronically connect networks operating at dissimilar security classification levels and supports the seamless, high-speed, controlled flow of information across security domains. This version provides significant increases in performance (an increase in throughput rate from 1.3 MB/second to 3.4 MB/second, a standards based open systems oriented Guard application, and a more stable and secure trusted interface). Future plans include hosting the Guard application on a Trusted Solaris platform and migration to Defense Messaging System (DMS) functionality.

### *Current Access to Data Repositories*

In today's environment, the ability to gather information on a "particular item of interest," requires a user to log into each of the data sources using a unique interface for each source. Once logged in, the user must be cognizant of the interface for the source they are using to gather information. While the concept appears straightforward, it requires the user to be very knowledgeable with the various systems. In addition, it requires developers to spend an enormous amount of time and money designing/implementing these unique client interfaces for each data source. Improving data access is the mission of Broadsword.

Broadsword is a secure web-based application, which provides improved access to Department of Defense data repositories. Broadsword provides users simultaneous access to multiple and geographically separated data sources through employment of a web browser. Broadsword accomplishes this by deploying middle-ware translators know as Gatekeepers. These Gatekeepers take a single, simple or complex search criteria from a Broadsword user and translate this search criterion into native queries supported by legacy data sources. Gatekeepers then consolidate the responses of the disparate data sources into a single response, and return it to the Broadsword user. Broadsword is designed for employment across the spectrum of operational and security environments.

Broadsword especially benefits users at locations lacking robust command and control, communications and intelligence resources. Leveraging the power of web-based computing, Broadsword users need not be at the same physical site as the data repositories they need access to. Instead, Broadsword provides users the ability to access data remotely by accessing their home unit Gatekeeper and retrieving only the latest updates. Conversely, rear-echelon analysts using standing profiles and file transfer protocol (FTP) processes between Gatekeepers may push the data.

Broadsword employment is tailorable to organizational mission, unit function, and personnel task level. Broadsword's core ability is to improve data access.

### *System Certification and Accreditation*

Generally, US coalition computer-based information systems and LAN are designed and developed to meet the Secret and Below Interoperability (SABI) requirements and all local site accreditation requirements as set by the Designated Approval Authority (DAA). Based on guidance from the project's SABI Customer Advocate, a proposed system should then anticipate the SABI/accreditation process to include system profiling by National Security Agency. If the proposed technology has not been previously approved as a SABI Reference Implementation (SRI), than the development will be executed as a New Technology effort under the SABI process. Detailed guidance for system certification and accreditation effort would then be provided by a SABI Customer Advocate team. It should be readily apparent that any program that qualifies for the SABI process and a SABI Customer Advocate would be time consuming and manpower prohibitive to design and develop.

The SABI requirements could call for the use of the tailored SABI version of the DoD Information Technology Security Certification and Accreditation Process (DITSCAP), Systems Security Authorization Agreement (SSAA). The SSAA includes two parts, an Executive Summary and a list of appendixes that are the technical and policy documentation for the C&A process. In addition to the documentation specified in the SSAA, the proposed system would have to recommend the following technical requirements and design documentation:

- Security Policy
- Functional Concept of Operations (CONOPS)
- System Functional Requirements
- System Functional Specification
- System Top-Level Architecture and Design

A detailed Coalition CONOPS should be written in the early planning stages of any experimentation. The CONOPS should document the assumptions and general approach to be taken toward creating that specific coalition environment. The CONOPS should become part of the accreditation package.

No means have yet been identified to permit collaboration on a data network across security domains. A real attempt to build an open floor CAOC with non-US personnel as full partners will require significant funding and an increase in staff to implement disclosure and security policy.

To maximize the chance for success, timelines for obtaining accreditation should be observed. DAA initial accreditation actions normally begin not later than 150 days prior to network activation. Non-standard security solutions (firewalls/routers, new guards) cannot be implemented in a short timeframe. Extensive testing by national agencies requires long lead times. Long-term funding could be a consideration for this type of large-scale effort.

## Current Security Architecture Challenges

### *Highest Common Security Level Network*

The US Command and Control ($C^2$) computer system, Theater Battle Management Core System (TBMCS), will be used as an example to illustrate this type of network. Unfortunately, US $C^2$ computer systems in general cannot easily support coalition operations due to disparate data security requirements. However, one option when using a $C^2$ system like TBMCS would be maintaining required databases, producing message sets and executing coalition operations on a ‘highest common security level network’, i.e. the highest classification/level data that can be contributed by each partner without violating their laws and regulations.

There will be certain functions (such as the plans and locations of individual nation's special operations forces or stealth aircraft mission planning requirements) that require individual nations to procure, manipulate and disseminate data separately from the coalition network to avoid inadvertent disclosure of sensitive data as decreed by law or regulation. There is an inherent risk associated with this and it will be covered in the next section.

### *Acceptable Risk?*

There are risks that must be addressed and accepted when a nation attempts to reduce the chance of compromising national secrets and capabilities by operating autonomously in a coalition environment.

An example will be given. "Nation X" has an aircraft whose mission planning requirements require the use of data that is classified "Nation X Only" or perhaps the mission planning process exposes certain vulnerabilities of the weapon system that "Nation X" does not want disclosed to certain members of the coalition. To protect it's asset and subsequent data, all mission planning is conducted in a "Nation X Only" environment or enclave, separated from the rest of the coalition partners.

Let us assume this aircraft requires support from coalition resources (aerospace, tankers, EW support aircraft, etc.) to conduct it's mission. The "Nation X" enclave is plugged into the coalition network on a separate terminal, has access to this information and uses it in it's mission planning.

Unfortunately, the folks who are doing the coalition planning and monitoring the mission execution of their assets do not know the support requirements of "Nation X's" aircraft and have modified the route, time on station or coverage of the support assets that "Nation X's" aircraft is relying on to execute it's mission. Because the planning for this mission has been executed outside the coalition environment, "Nation X" has put at risk the very assets they are trying most to protect.

On the positive side, there are workarounds utilizing a TBMCS-based coalition network, but these require additional manpower. The required additional manpower is diametrically opposed to the reduced footprint goals we have established to lessen our force protection vulnerabilities.

### *Highest Security Level Network*

Today, the preponderance of stealth or special mission aircraft and their associated 'special access program' mission planning requirements and vulnerabilities belong to the US. Also, most of the collection platforms are classified 'US Only' whose capabilities are guarded by US law and regulations. Because of this, the operational commander may select the option to have master data classified as US Only. Generally, this option requires significantly more US manpower. If the $C^2$ databases are on the US Only side, the full functionality of TBMCS can be used and information within the individual applications can be populated with the most precise 'US Only' classified data available. Current technology will allow the ATO/ACO and other required message sets to be produced and delivered to coalition partners. Unfortunately, this can lead to mistrust among coalition partners – especially those that don not have similar assets and restrictions placed on what they can share in a coalition environment.

### *No Databases*

The third option that could be considered by the operational commander, with due diligence, is not to populate the large $C^2$ organic databases. This option would be necessary in the case that there is not enough time to adequately train, or manpower (coalition or US) available, and systems are non-available. US $C^2$ systems are designed to operate in a joint environment with many processes going on unseen by the operator. Example: In TBMCS, to populate the AODB with friendly fire units so the operator can see the location of the units when the Friendly Order of Battle (FROB) is queried, TBMCS must receive a B220 message from the Army's Advanced Field Artillery Tactical Data System (AFATDS).

The no database option is measurably less efficient due to the fact that improvised procedures must be developed and learned by all partners. This "fall back" position does have the advantage that generally Commercial Off The Shelf (COTS) software is used in conjunction with Management Information Systems (MIS) products on a common LAN. This methodology is not the most efficient, but it allows coalition partners to be engaged in the planning and execution from the start of the operations. Operational Commanders should be completely aware of all consequences and limitations caused by selection of this option, including the significant increase in manpower required.

In any operation there may be a mix of all three options as the Commander attempts to mitigate risk and improve combat effectiveness.

## Experimental Methodology

To achieve greater opportunities for enhanced mutual interoperability and capability, liaison elements from both the Coalition and US Forces should operate in a collocated area within the area of responsibility (AOR). The composition and functionality of the US initial cadre operating in the AOR, working with their Coalition counterparts, should be responsible for developing processes and procedures to implement integration/interoperability initiatives validated through experimentation.

Moreover, a Coalition liaison element comprised of 20 – 30 personnel should operate from the US, dispersed to appropriate functional areas to operate "side-by-side" with their US counterparts.

### *EXPERIMENTAL METHODOLOGY (Coalition Information Architecture)*

In December 1999, Air Staff and the Defense Intelligence Agency approved the requirement to allow high-side (a more secure network) war fighters to see low-side sources (a less secure network) (e.g. Air Order of Battle Database (AODB)), query them and request products to be delivered to the high-side user (e.g. Situation Assessment Analysts). Coalition Information Architecture (CIA) should expand this requirement to allow a reverse (reach up) capability.

CIA could act as the umbrella initiative to provide automated multi-level security data sharing to support coalition interoperability between Joint World-wide Information Communications System (JWICS), SIPRNET, NIPRNET, a Coalition Wide Area Net, and theater specific LAN. Functionality should reside within the CIA infrastructure without needing to build an independent infrastructure.

This capability should augment existing $C^2$ systems, not replaces them. CIA encompasses the benefits of Broadsword and ISSE Guard by adding the Trusted Transfer Agent (TTA).

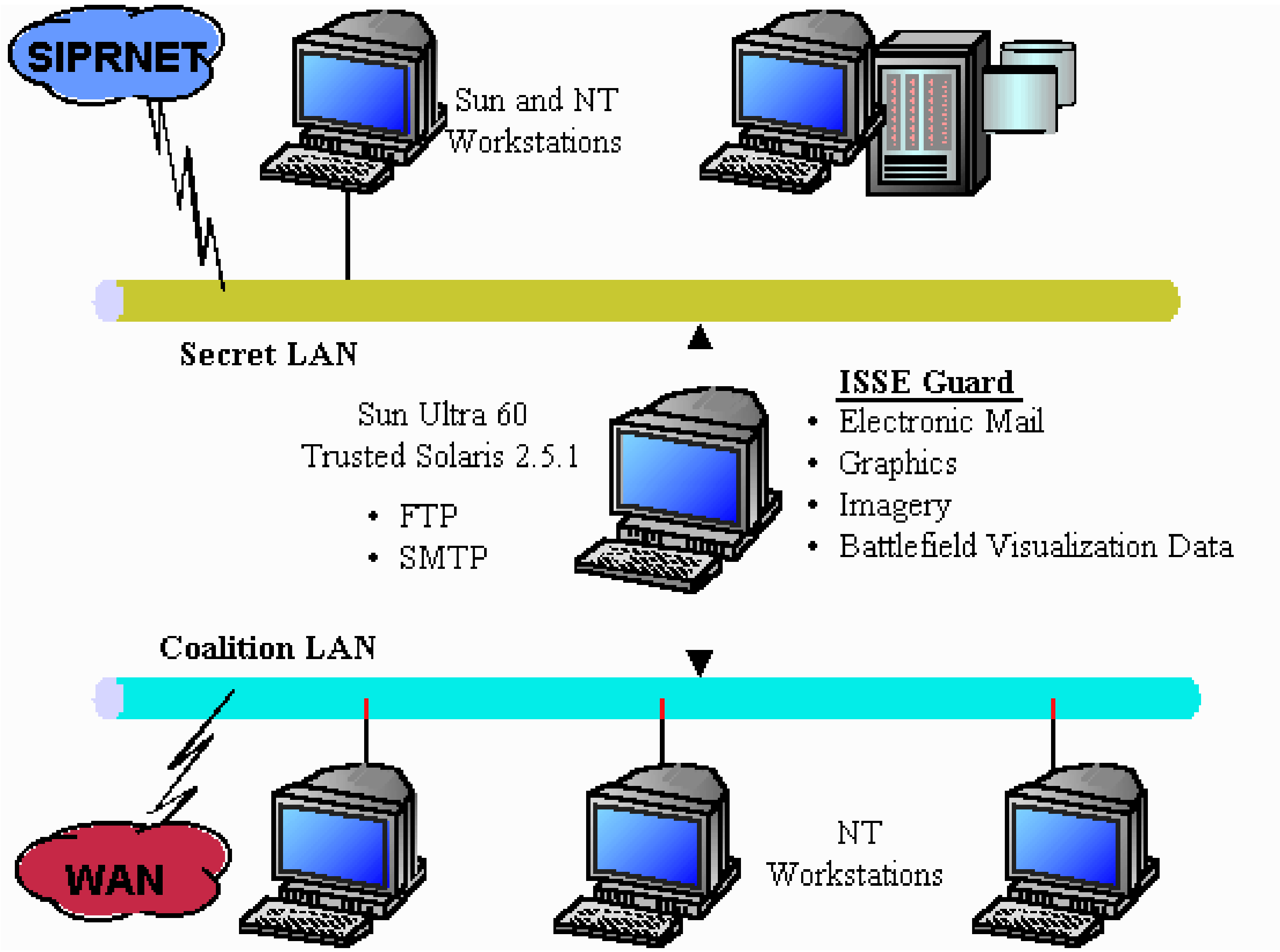



**Figure 1:** ISSE Guard Functional Architecture

The TTA program leverages the strengths of Broadsword and ISSE Guard to enable Multi-Level Security (MLS) information access. TTA brings together this powerful infrastructure and the multiple security level capability provided under the ISSE Guard. In order to ensure that high side information is not inadvertently passed through the TTA and ISSE Guard to the low side, two levels of extensive security filtering capabilities are provided: message level filters and field level filters.

Messages level filters use a “dirty word” list containing a list of words and/or phrases that are either not passable to the low side (e.g. classified code words, etc.) or strong indicators that the associated information in the message is not passable to the low side (e.g. security labels). By applying the message level filters it is determined if a message being passed through the TTA (and subsequently the ISSE Guard) from high to low contains any “dirty words”. If a message is found to contain one or more words/phrases in the dirty word list, the processing of the message is terminated. Field level filters are an additional capability added to the TTA. Since the messages passing from high to low through the TTA contain formatted field-value pairs, additional filtering can be provided on a field-by-field basis. For each field within each message type, over which field level filter is needed, an entry in a file is generated describing how the information in the field is to be filtered. A variety of filter types have been created which test for conditions such as Value in Field, Value Not in Field, Value in Range, etc.

When individual applications like ISSE Guard, Broadsword, and TTA are used on CWAN, users from various countries can become one effective fighting force. These users will be able to exchange e-mail messages with the Secure Internet Protocol Router Network (SIPRNET) users via the ISSE Guard. Additionally, Coalition users will be able to access various power point briefings, ATOs, and other large files via a web server located on CWAN. Web site content managers will replicate the web site residing on CWAN and SIPRNET web servers to corresponding web servers. This replication process will permit Coalition and US Forces access to identical web sites with content releasable to Coalition partners.

During operations, record message traffic could become backlogged, and the delay could create difficulties for coordination and execution. Message traffic could be posted to the local web server, thus providing much faster access. A word of caution – E-mail is a useful tool for coordination, but large e-mail attachments, such as an ATO being "pushed" to all Coalition tactical level users, could severely affect naval ships or other tactical users with low bandwidth capability.

One of the goals of the CAOC should be to provide access to Coalition information for all Coalition partners via a common web site. To achieve this goal, $C^2$ should be established on a CWAN that will link non-US combined forces to e-mail and web services hosted by the CAOC. A secure mail guard (e.g. ISSE Guard) should allow US personnel to access e-mail and other web services via SIPRNET, and at the same time allow information posted to the common web site to be releasable to all coalition partner countries. Current web site technology should be used to distribute and collect operational information.

The web site should function as a "digital binder" and contain planning documents such as Rules of Engagement (ROE), Schedule of Events (SOE) and periodic reports/orders such as Commander's Intentions, Operational Reports (OPREPS), and standard United States Message text Format (USMTF) message sets like the ATO/ACO. The Coalition Commander would provide guidance to his forces through documents posted on the common web site. These documents could be in any form that can be displayed on a web site.

The goals of the Coalition Web Site should be two-fold:

1. To provide intelligent, timely and relevant information and knowledge on operations and intentions from subordinates to superiors and from Tactical Force Commanders to the Coalition Commander.

2. To provide a means for all military forces to exchange significant tactical and operational information in hopes of ultimately replacing message traffic.

**Interoperability Opportunities**

Operational processes that have a potential to be readily integrated are: air picture, theater missile defense notification, ATO integration (common message sets), secure communications systems and data networks, intelligence distribution, and common weather picture.

***Common Operating Picture***

The term, Common Operating Picture or COP, is one of the most misused words in today's $C^2$ vocabulary. As a result, $C^2$ warfighters, staff representatives, and technologists frequently face a great deal of confusion in trying to discuss the concept, employment, utility, and evolution of the COP. Much of this confusion can be resolved through reference to Chairman, Joint Chiefs of Staff Instruction (CJCSI) 3151.01, Global Command and Control System Common Operational Picture Reporting Requirements. It defines COP as an information tool supporting the warfighting CINC. It further defines the Common Tactical Picture (CTP) as an information tool supporting the Joint Task Force. In general, CJCSI 3151.01 describes the COP/CTP as a tool that provides battlespace awareness through a variety of information domains including, "current, anticipated, or projected, and planned disposition of hostile, neutral, and friendly forces." The COP/CTP may also depict logistics, readiness, planning, as well as other data to support Commander-in-Chief and coalition operations. Although the instruction makes reference to Component participation in feeding/maintaining the COP/CTP environment, it does not define or describe how the Components will participate in that activity.

(US) TBMCS/SAA and (Coalition) TBMCS/SAA could be dynamic tools providing users and COP Masters/network administrators options in supporting higher headquarters commanders and their subordinate units. (US) COP/CTP would provide every user within a designated network the ability to see the same Near Real Time (NRT) picture. (Coalition) COP/CTP would provide every coalition user within a designated network the ability to see a limited NRT picture. In doing so, it also provides each participant the ability to update, add to or corrupt the NRT picture the entire network (AOC on down) sees. Given this, COP Masters must implement a stringent set of guidelines and procedures. Most importantly, COP Masters must ensure that each participant in the COP is thoroughly trained on the system. COP/CTP Masters must establish and enforce a set of rules that balance user requirements against the COP Masters' need to ensure against data loss and/or corruption. A process of database synchronization is used to accomplish the transmission of data to the COP on a single LAN.

RADIANT MERCURY is a software application developed under contract to the Navy, which can automatically sanitize and downgrade formatted classified documents and synchronize a (US) COP/CTP with a (Coalition) COP/CTP. RADIANT MERCURY operates according to predefined security rules. The automation of the sanitization and downgrade process decreases the time needed to perform these functions, and eliminates human error.

Once an authorized node is brought into the COP/CTP, the entire track database is copied from the SAA (US) or (Coalition) server onto their nodes Track Data Base Manager (TDBM)(US) or (Coalition). New reports and track updates received by the COP/CTP Master are copied onto client nodes as the tracks come in from NRT sources. Updates, deletes, and modifications to data in the COP from client nodes are copied to the COP/CTP Master server/terminal at regular intervals.

However, the technical solution once implemented, will provide a common operating picture to the maximum extent possible that each country authorizes for release. Viable integration and interoperability of US and Coalition air pictures, procedures, and equipment can be achieved through the following:

1. The US should provide personnel in the CAOC to work with Coalition counterparts to reconcile dual tracks and other link management issues.
2. Coalition members, on a case-by-case basis, should jointly produce documentation referencing operational procedures for the combined data-link network.
3. US Forces and Airborne Early Warning aircraft (E-3 and E-2) can use Coalition sites as their primary TADIL-A ground entry station (GES); but other GES locations may be used as necessary.

The term Air Component Picture (ACP) is defined as the integrated battlespace picture derived from sources managed by the Aerospace Component. As a subset of the CTP and COP, it establishes the aerospace portion of the COP's battlespace infosphere. The ACP data may contain real-time or near real-time battlespace information; aerospace component planning information; aerospace logistics and readiness information; as well as relevant information distributed down from the C/JTF CTP or horizontally from other component's pictures. When properly managed, the ACP will provide dominant battlespace knowledge to synergize the command and control of the Aerospace Forces in support of combined operations.

The ACP is a powerful information tool, providing access to several information domains pertinent to aerospace operations. Readers should note that the COP, CTP and ACP software are virtually identical. The difference in the three terms include level of employment (e.g., strategic, operational, tactical), focus on organizational functions, and the manner in which the information is processed for decision-making.

### *Theater Missile Defense Notification*

Theater Missile Defense (TMD) functions, within the defensive operations branch in a CAOC, fulfill the roles of passive and active theater missile defense. Additionally, this function works cooperatively with the Intelligence to determine enemy theater missile areas of activity/interest for further exploitation. The TMD function may have space and intelligence liaison representatives manning specialized equipment consoles for passing missile launch and impact point alerts. They also search for areas of interest for further reconnaissance exploitation and possibly eventual creation of either a time critical or standard target process.

The defensive operations branch is separated into three primary interconnected functions. These functions are air-breathing threat for fixed and rotary wing assets, theater missile defense for passive and active defense, and link management control, to provide an accurate consolidated air picture. Air Breathing Threat function is what many people think of as the traditional role of air defense. This function manages defensive fighters and surface to air missile defenses in direct support of the air defense mission. Defensive Duty Officer (DDO) evaluates and recommends changes in air defense activity, airspace management, and surveillance performance. He sits on a console displaying the air picture and remains in contact with subordinate units. Duty officers for specific coalition aircraft or missions may be available to assist in coordinating defensive operations. The DDO with the help from air defense liaisons coordinates air and missile attacks against threatening enemy air targets. The DDO could potentially receive time sensitive information from a common voice net (Voice over IP) that could be used for tactical dissemination and response.

### *Standard Message Sets*

The Defense Information Systems Agency (DISA) is responsible for maintaining US Department of Defense information technology standards and conventions. Within DISA, the Center for Standards is the designated configuration manager for the United States Message Text Formatting (USMTF) Program. The USMTF Program documentation consists of two major documents, MIL-STD-6040, United States Message Formatting Program, and CJCSM 6120.05. To use military diverse forces effectively, a continuing need exists to increase the fighting capability through compatibility among the various C2 information systems and interoperability at the information level. This is done through the adoption and maintenance of standards and systems designed to provide interoperability through the use of approved joint data and information exchange standards. The operational benefits gained by allied forces from the use of common or compatible C2 standards are essentially identical to those for U.S. forces engaged in joint operations. In the combined environment, an additional advantage is the alleviation of information exchange problems associated with differing national languages and military organizational structures. Because of the large United States investment in tactical C2 systems the USMTF is the most logical interoperability standard that should be used in message sets. An example of two USMTF message sets used in the CAOC would be the Air Tasking Order (ATO) and the Air Control Order (ACO)

### *Electronic Mail (E-Mail)*

Let us examine how to handle e-mail in a coalition environment. It is often impractical to maintain a message transport system on certain types of smaller C2 nodes. For example, a workstation at a tactical unit may not have sufficient resources (cycles, disk space) in order to permit a Simple Mail Transportation Protocol (SMTP) server and associated local mail delivery system to be kept resident and continuously running. Similarly, it may be expensive or impossible to keep a tactical personal computer interconnected to an IP-style network for long amounts of time (the node is lacking the resource known as “connectivity”). Despite this, it is often very useful to be able to manage mail on these smaller network nodes, and they often support a user agent to aid the tasks of mail handling. The Post Office Protocol – Version 3 (POP3) is intended to permit a workstation to dynamically access a mail dropped on a server host in a useful fashion. Usually, this means that the POP3 protocol is used to allow a workstation to retrieve mail that the server is holding for it. The well understood commercial standard of POP3 would most likely be used during any coalition operation. The problem that needs to be understood is difficulty in transferring e-mail between networks of various levels of security.

A secure coalition e-mail system generally utilizes a high assurance guard system (ISSE Guard) to permit exchange of e-mail without attachments between dial-in users and other units on the CWAN and SIPRNET. This service could use a SMTP/POP/MS EXCHANGE server requiring a compatible e-mail client. The coalition web site could be hosted on web servers in the CAOC. To facilitate browsing by all forces, the web site would be replicated from SIPRNET web servers to Coalition web servers. Updates to the web site would be conducted as operational requirements dictate. Do to the use of https services, browser would require a current version of MS Internet Explorer V5.x or higher. The CWAN should support the exchange of classified files between all units utilizing an FTP server. These files would not be accessible to SIPRNET users, but could be transferred from the CWAN to SIPRNET and vice versa by trained staff system administrators. Web clients are acceptable programs for accessing the FTP services. CWAN policies would insure all e-mail messages transmitted from the classified SIPRNET to the classified CWAN and vice versa contain no attachments. Only e-mail with information classified releasable to Coalition users should be transmitted. Generally the producers of information are responsible for reviewing and approving the release of e-mail information prior to transmission. All data, services, and other controlled resources should be protected from unauthorized use. Users should have access only to data, services, and resources for which they have the clearance, authorization, need-to-know, and need-to-use. All users should be identified and authenticated before the users are granted access to data, services, and resources.

### *Intelligence Distribution*

The CAOC Director of Intelligence is responsible for all intelligence operations in support of coalition ATO/ACO preparation and execution, including support for all intelligence organizations within the CAOC framework and at subordinate tactical units.

Responsibilities could include intelligence exploitation, fusion, and targeting operations. Additionally, they could also include the dissemination of intelligence information to the Coalition Air Component Commander staff, combat operation cell; combat plans cell; and subordinate tactical units to produce one commonly understood picture of the battlespace.

One of the major functions of intelligence is to maintain and distribute the current Order of Battle. Most likely in any theater the Commander-in-Chief (CINC) will be responsible for maintaining General Military Intelligence (GMI) information within his area of responsibility (AOR). The CINC will maintain this information in an MIDB server. The CINC organization could provide the same MIDB files to other key command control systems in the AOR. The CINC could also establish a dedicated GCCS-I$^3$ ISD server to provide the data extract to CAOC systems like TBMCS. Any operational or tactical unit could receive updates directly from this site. The CAOC should maintain a current tactical threat Order of Battle to support campaign planning and mission execution.

A secondary function of CAOC Intelligence is imagery storage and analysis. Imagery can be categorized into two basic formats: National Image Type Format (NITF), which is the future standard; and what may be referred to as a common workstation format (e.g., GIF, MIF, TIFF, and Sun Raster). The NITF format is complex and can contain multiple images, header data, symbol data, and unique data extensions.

One of technological ways that allows sensitive digital images to be tracked is to insert a covert digital watermark. The digital watermark would stay with that image, even as it is copied, altered and distributed. The digital watermark becomes an imperceptible part of that image, helping maintain an important link between the original image and any derivatives of that image. Watermarking will play an important role in enhancing the security of individual national states digital assets as they engage in daily CAOC functions.

### *Communications/Information Management*

Coalition communication/ information management could include the following common functions as a minimum:

1. Track plain language addresses (PLA) and routing indicators (RI) to ensure message routing currency.
2. Ensure a common phone book is developed and distributed to all coalition members.
3. Coordinate fix actions on lost/delayed/misrouted message traffic.
4. Ensure a single distribution point for all incoming and outgoing messages. Direct alternate routing if a backlog develops.
5. Monitor ATO/ACO transmission to help ensure receipt and work with Combat Plans to resolve ATO transmission and receipt problems.
6. Build web pages and manage web information and shared folders.

### *Common Weather Picture*

The CFACC will likely modify the presentation of the detail content of the weather briefing after the first few briefings. Generally the strategic or regional weather forecast will be on a less than SECRET LAN's web page along with a current satellite picture that covers the area of the AOR. Collocated with the web pages should be the official Horizontal Weather Depiction (HWD) forecast and other HWDs from other regional or strategic forecast centers. Based on the HWDs and the official Terminal Aerodrome Forecast for the next 24 hours, information about Coalition Airfield Weather, tactical locations weather, coalition weather forecasts out through 48-72 hours could be produced. High Frequency and Ultra High Frequency space weather information could be used to predict the Communication Impacts Due To Space Environment. Products that could be shared include:

1. Current Hemispheric or Global Meteorological Satellite (METSAT)
2. Current Regional or High Resolution METSAT
3. 0-24hr Cloud Cover And Weather
4. 24-48hr Cloud Cover And Weather
5. Coalition Airfield Weather 00-48hr
6. Communication Impacts Due To Space Environment, as required.

## Conclusion

In conclusion, a road map has been proposed to support coalition integration and interoperability of information systems within a U.S. run Coalition Aerospace Operations Center. It is aimed to be used by personnel to explore how Command and Control systems (One set of systems on a US secure network and one set of systems on a coalition LAN) could work during experimentation. Military members (both US and Coalition), Contractors, acquisition, developmental test organizations, operational test agencies, and operational users are the key during C2 events and should document procedures and workarounds that improve overall CAOC system. This paper is not designed to provide standard operating procedures for any standing Coalition Aerospace Operations Center. Primarily, it is to provide operational users some insight into using more than one suite of equipment to prosecute a Coalition Air Operation.

This paper expresses in both operational and technical language the theoretical underpinnings required for experimentation to improve a notional Coalition Aerospace Operations Center.

---

**References:**


1. 1. Jon L. Boyes and Stephen J. Andriole, eds., *Principles of Command and Control*, Foreword by Gen. Russell E. Dougherty (Fairfax, VA: AFCEA International Press, 1987).
2. 2. Eric M. Grose and Mark W. Smith, “From Military Command and Control to Bond Trading: The Human Factors Process,” in *Proceedings of the Human Factors Society 36th Annual Meeting, SYSTEM DEVELOPMENT: Extending Military Human Factors* 2 (1992).
3. 3. Melissa A. Cook, *Hewlett-Packard Company. Building Enterprise Information Architectures, Reengineering Information Systems* (Upper Saddle River, NJ: Prentice Hall PTR, 1996).


---

**MARVIN L. “LENARD” SIMPSON, Jr.** has a BS in Mechanical Engineering Technology from Virginia Tech University and an MS in Administration from Central Michigan University. He is now a Defense Contractor (L3 Com., Analytics Corporation) engaged in direct support to the Air Force Command and Control, Intelligence, Reconnaissance, and Surveillance Center led by Major General Robert F. Behler, after having completed 20 years as a member of the United States Air Force. His operational background includes experience in air-to-air and air-to-ground fighter missions, including the use of “special weapons,” leading operational support teams, exercise planning, and over 9 years in command and control informational systems. He is a recognized expert in the use of the U.S. Air Force’s Theater Battle Management Core System (TBMCS) and is familiar with most currently fielded Command and Control equipment. His technical skills include UNIX System administration, MCSE (CORE plus TCP/IP and Exchange), and PKI. He is a Commercial Pilot with more than 1450 flight hours in, F-4, T-43, T-37, T-38 aircraft and in general aviation aircraft.

**BACK TO TOP**

---

# Integration and Interoperability of Information Systems within the Coalition Aerospace Operations Center

*Marvin L. "Lenard" Simpson, Jr*

**Keywords:** Coalition operations, interoperability, coalition command and control, C4I, coalition aerospace operations center, CAOC, CFACC, TACS, coalition ATO, airspace control authority

**Abstract:** This paper presents a roadmap to support coalition integration and interoperability of information systems within a Coalition Aerospace Operations Center (CAOC), run by the US. The proposal may be used to explore how C2 systems (one set of systems on a US secure network and one set of systems on a coalition LAN) could work during experimentation. Military members, both US and Coalition, contractors, acquisition, developmental test organizations, operational test agencies, and operational users are the key during C2 events and should document procedures and workarounds that improve overall CAOC system. Instead of presenting standard operating procedures for any standing Coalition Aerospace Operations Center, this paper primarily provides operational users with some insight into using more than one suite of equipment to prosecute a Coalition Air Operation. This paper expresses, both in operational and technical language, the theoretical underpinnings required for experimentation to improve a notional Coalition Aerospace Operations Center.

**full text**

Authors: **Javier Carbo, Jose M. Molina and Jorge Davila**
Title: **Argumentative Negotiations with Anonymous Informer Agents**
Year of issuance: **2002**
Issue: **Information & Security. Volume 8, Number 2, 2002, pages 194-206**
Hard copy: **ISSN 1311-1493**

---

# ARGUMENTATIVE NEGOTIATIONS WITH ANONYMOUS INFORMER AGENTS

Javier CARBO, José MOLINA and Jorge DAVIL

---

**Table Of Contents:**



---

## Introduction

Internet brings together people geographically and culturally unrelated to each other. This is one of the most remarkable features of electronic communities. In such context there is an extended sense of anonymity where trust and confidence might take place easier than in real life. Information Agencies may benefit from this feature of electronic communication in their endeavour to acquire secret information from previously unknown sources.

All this can be achieved through the use of autonomous software agents. However, the success of agents has not met the expectations. Among the possible reasons, we could mention the fear of users of being cheated, and of revealing unconsciously their personal opinion and preferences.

Users would feel comfortable delegating their duties to agents providing the behaviour of these agents exhibits intelligence, in other words, if the agents act rationally and could justify their decisions. Intelligence in negotiation is used to consider and evaluate offerings according to the particular preferences of the user. Nevertheless, success in negotiations does not depend only on the details of each offering. A certain number of arguments can be used to persuade the other party in the negotiation.

In security and military scenarios, the information that could be collected about an informer has a strategic value. For that reason, informers should avoid revealing unnecessary information, and a certain level of ambiguity is required. They can achieve it if they could reason with arguments and at the same time keep the anonymity of the informer.

Negotiations take place with the aim to persuade potential informers offering certain compensation (money, new identity, etc.). Agents may play a relevant role in this process since the potential informer would be able to negotiate in a rational and quick manner with several intelligence agencies at a time and a customised procedure for this could be devised.

**Previous work on negotiation**

Negotiations take place by the exchange of messages that aim at reaching an agreement satisfactory to both parties. Negotiations have been extensively studied in Game Theory.[1] The major issues in any negotiation are protocols and strategies.[2] Protocols rule the communication acts allowed at each moment of a negotiation. They should be public and accepted previously by both parties. Strategies rule the particular behaviour of each agent in a negotiation. They are private because they have to reflect the personal preferences of the user represented by the agent.

Game Theory assumes that the negotiating parties have complete knowledge and total rationality. The former means to know the preferences and beliefs of all the participants, the latter means the ability to reproduce the computations of any other participant. When we study negotiations from an agent-mediated perspective,[3] these two assumptions are too restrictive, especially in open and dynamic environments of heterogeneous – and possibly malicious – agents.[4]

The alternative provided by the adoption of software agents is based on a shared ontology of universally accepted terms in the domain;[5] only the protocols are publicly known, and the strategies are not optimal but computed in real time. So the protocol has a strong influence on the strategies adopted by the agents.[6]

The protocols are characterised by their cardinality, the environment, the negotiation issues, the temporal requirements, and the attitude of the participants.[7] These parameters establish the space of possible modes of acting. The larger the negotiation space, the more probable is the agreement. At the same time, this entails an increase in communication and computational costs. Considering these characteristics, different types of negotiation processes can be distinguished.

Among them, auctions have become the most popular type of electronic negotiation. They are usually focused on a single issue (price), with cardinality 1-to-n and a very complex and predetermined interaction. The number of alternatives that can be considered is very limited; hence the results of the negotiation are restricted.[8] Some researchers do not regard auctions as a negotiation.[9] Further, simple protocols, such as Contract Net, have also been proposed.[10] Here, an agent could simply accept or reject totally the offering of the merchants. Provided the user agent is able to propose counter-offerings, a greater level of sophistication can be achieved.[11]

However, auctions are very different from the daily bargaining at markets. Most people are not familiar with their rules. If agents intend to reflect the real behaviour in a society, then a human-like negotiation would be needed.[12] Humans explain and deliberate rational arguments to persuade the other party to improve its offering. This is often called persuasive argumentation in negotiation.[13] These arguments are represented with illocutions. Researchers in psychology have studied several illocutions involved in persuasive negotiations, for instance threats, rewards, and appeals[14]. These types of arguments were applied to the domain of labour negotiations by Sycara [15] and formalised in the works of Sierra *et.al.*[16] and Kraus.[17]

**Security Services and Mechanisms**

The security architecture of communications considers the following services achieved by cryptographic mechanisms: privacy, authentication, non-repudiation, integrity, and access control.[18]

1. Privacy provides the confidentiality of data exchange between entities. An encryption algorithm can provide this protection, making it computationally impossible for unauthorised users to access the information.
2. Authentication ensures that an entity is not supplanted by another. When a digital signature is used for providing authentication, then it is known as “strong authentication”.
3. Non-repudiation can be provided by the source or by the destination in communication. In the first case, non-repudiation of sending, the sender of the message cannot repudiate its emission. On the other hand, non-repudiation of receipt implies that the receiver of the message cannot deny getting it.
4. Integrity supposes the detection or prevention of the unauthorised modification of the information. This service is aimed to ensure that the information is correct and complete.
5. Access control is frequently confused with the authentication service. However the access control is in charge of providing the appropriate privileges to the users.

From the group of cryptographic mechanisms used to provide such services, we describe the general properties of symmetric and asymmetric encryption, and hash functions. Interested readers can gather more information on these issues in Schneider’s book.[19]

Symmetric (also called conventional) encryption has been largely used from the very beginning of cryptography. This type of ciphering provides balance between security and performance. However, it is necessary that the parties involved agree on a secret key through a secure channel. This is a clear limitation of this type of encryption due to the fact that in open and populated systems it is not an easy task to establish secure channels between parties that do not know each other.

On the other hand, asymmetric encryption relies on a pair of related keys, one of them is secret (only known by the owner) and the other is public (which is known, ideally, by everybody). The private key is used to decrypt (or sign), while only the public key can encrypt (or verify a signature).

The main advantage of asymmetric cryptography compared to symmetric cryptography is that it does not require a secure channel to exchange keys. An entity can get privacy of its messages just ciphering those messages with the public key of the destination entity. In this way, only the owner of the private key will be able to decrypt such messages.

Although slower than symmetric encryption, public key cryptography is preferable when dealing with very populated systems. However, every asymmetric encryption scheme needs a complete infrastructure to certify public keys and to protect the private keys. This infrastructure, denoted PKI (Public Key Infrastructure), involves a considerable number of resources that sometimes are not affordable by information systems. In other cases the investments needed to implement a PKI cannot be outweighed by the benefits that can be obtained. In such cases the asymmetric cryptography is not used and, therefore, other cryptographic mechanisms – with lower cost – are chosen.

A hash function is a mathematical transformation that takes as input a set of bits of variable length and computes a short fixed-length output. This output has the important property that there is no other way to find what input produced it but the one of trying all possible messages. This feature of hash functions is used in some micro-payment schemes (such as Payword) to achieve origin's authentication by establishing a link between consecutive hash values, and the first one was digitally signed.[20] This approach is an example of how hash functions can be used to reduce the number of public-key operations and, therefore, increase the efficiency at the expense of relaxing security. In a similar manner, our system will take advantage of hash functions, this way reducing the computational cost of ensuring security.

### The Proposed Scheme of Negotiation

We intend to use informer agents that announce the knowledge of relevant secrets and that persuade intelligence agencies to improve their offering. In order to allow elaborated negotiations, the space of negotiation will be very open. To this end, we propose an extremely simple protocol: a sequence of offerings and argumentation-based counterproposals. The protocol follows the execution cycle described below:

1. The informer agent presents its credentials showing the knowledge of a relevant secret, but without revealing it.
2. The intelligence agency verifies the existence of such knowledge, and then it makes an offering to the informer agent.
3. The informer agent may then reject the offering sending a counterproposal with arguments to persuade the governmental agent. These arguments would take the form of rewards, appeals and threats. The counterproposal may also suggest what the agent expects to be improved.
4. Next, the two agents exchange offerings and counterproposal-based rejections sequentially.
5. Negotiation ends when the informer agent accepts the offering (an agreement is reached), or when any of the participants withdraws from the negotiation.

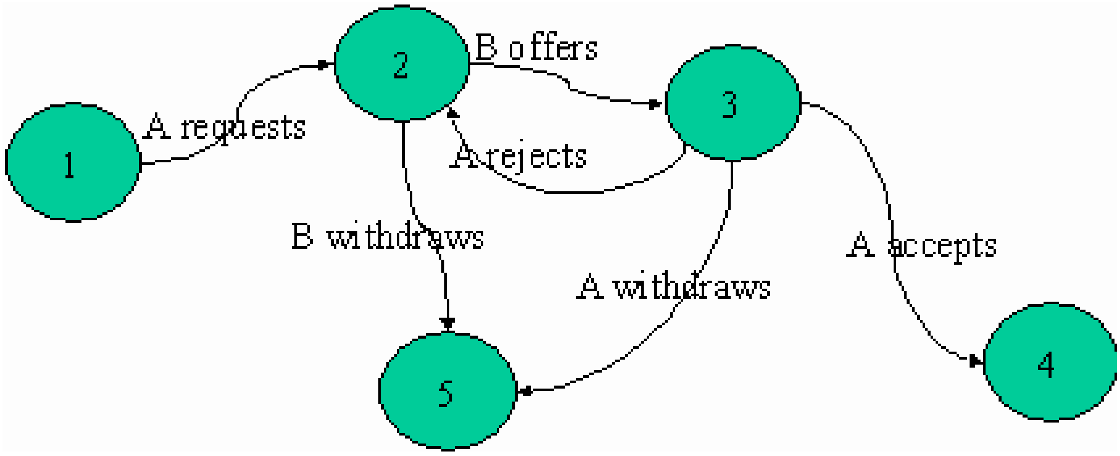



**Figure 1:** State Diagram of the proposed negotiation protocol

In order to reflect the expressiveness of human bargaining, each message has an illocution associated with the Speech Act theory.[21] These are: request, offer, accept, reject, and withdrawal. They will be codified using KQML (Knowledge Query and Manipulation Language).[22]

**Proving the knowledge of secrets**

The first step in negotiation is not trivial. We want an informer to present himself (herself) showing the knowledge of a certain secret without revealing the details of this secret. If the informer furnishes some information about the secret, it will not be fair due to the fact that exactly the secrecy of the information is the goal of negotiation, while, in response, the destination agent is not committed at all. In order to avoid such a disadvantage, we would like to protect the knowledge of the details regarding the secret information.

When the verification of the knowledge may be shown using algebraic properties, these requirements are analogous to the problems known as *two-party secure computations*. In them, both parties 1 and 2 know the definition of a function $f(x,y)$ . Each party knows just only certain information, $a_i$ , but not the information known by the other parties, $a_j \forall j \neq i$ . So, both parties want to know the output of the function $f(a_1, a_2)$ without revealing much information about their own secrets.

For example, Yao [23] applied public key encryption in a well known example of this particular problem: the millionaires problem, in which two players want to know who is richer, but they do not wish to disclose the exact amounts they own. Another proposed domain of application is voting scenarios were votes are secrets and some participants have veto ability.

However, the use of this technique is very limited. In these computations, the security of the solutions relies on certain algebraic properties that the adopted encryption scheme maintains. These encrypted outputs are homomorphic with the inputs, so they partially preserve the multiplicative group of operations (associativity, complementation, neutral element). This common algebraic structure is used to verify the existence of certain properties of the inputs without revealing them.

Previous work of the authors of this paper has presented a way to compute similarity between subjective shopping preferences without revealing the purchasing profile.[24] In them, we have applied the millionaires' problem to the four squares of the corresponding trapezium that represents a fuzzy set. Based on this idea, we could define a variant of a voting scheme where some questions to certain secrets are posed. Some of them will be known by the intelligence agency, other secrets are desirable to uncover. All the parties will know how many questions were posed by the other party, and how many of the answers coincided. As the secrets that form the questionnaire are proposed by both parties, the output of the function will give a fuzzy idea of the relevance and sincerity of the potential informer. The interpretation of such uncertain information will help intelligence agencies to estimate the quality of the potential informer.

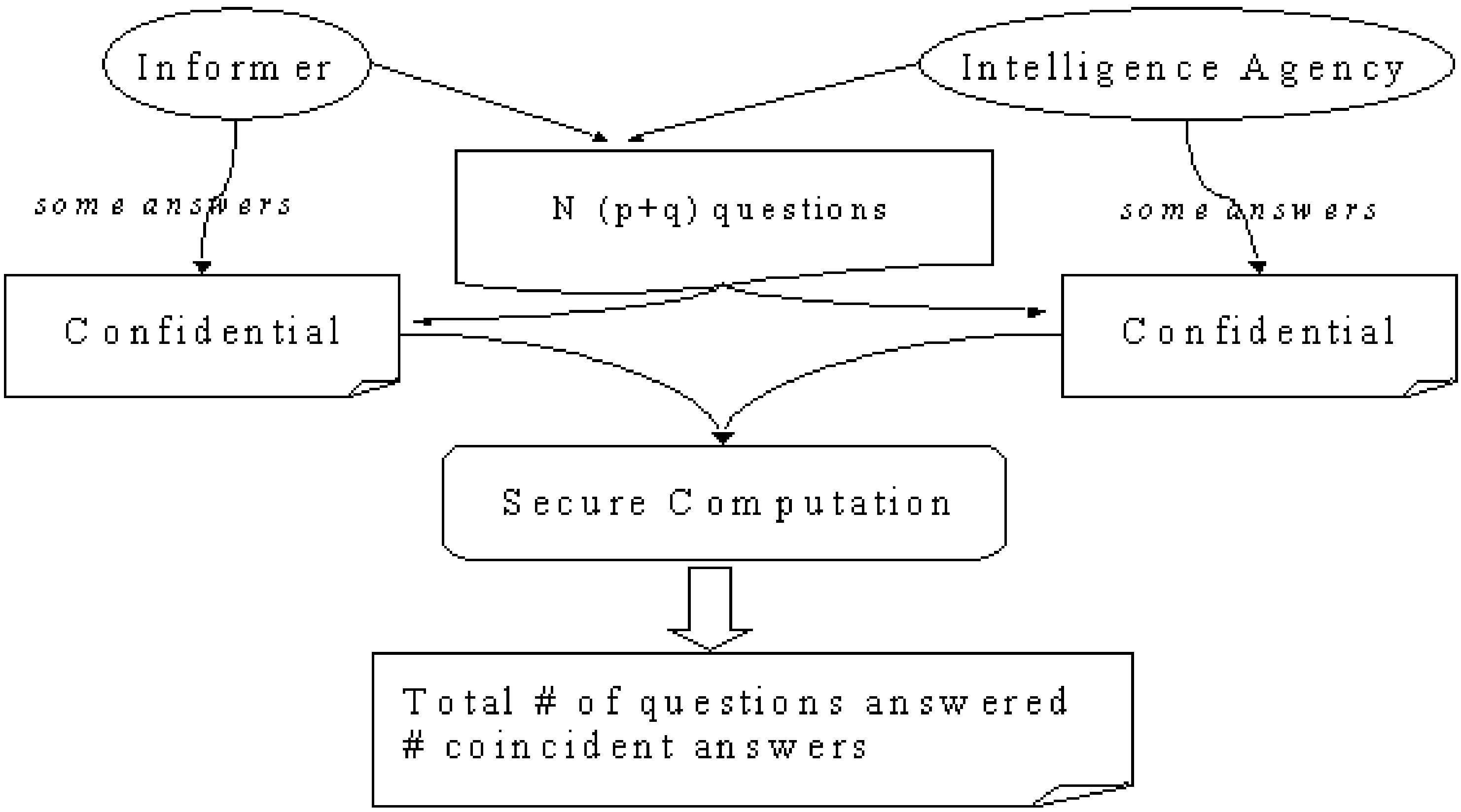



**Figure 2:** Sample introduction of informers.

**Exchanging arguments**

The attention in persuasive negotiations is focused on the representation of the arguments which support the counterproposals. In this paper we describe a subset of the possible rational arguments in negotiations of secrets between an informer and an intelligence agency.

One of the most frequently used arguments in such real-life negotiations are the threats of withdrawing from the negotiation. The threats, called ultimatums, are arguments of the type "take it or leave it." Deadline Time could be represented in two different ways: as a temporal limit, or as the number of messages to be exchanged in advance. Urgency would represent a critical factor in the strategy of negotiation. We can represent this argument as a threat by:

$$threat\ (a,\ b,\ \varepsilon\ ,\ [not]\ \psi\ ,\ t),$$

where *a* and *b* are agents, ε represents the desired improvement of the offering, ψ stands for the withdrawal used by agent *a* to threaten agent *b* at time *t*. This withdrawal would take place unless agent *b* accepts the improvements detailed in ε .

We propose another argument that is commonly used in such type of negotiations: possible rewards for revealing better secrets in the future. In other words, informer agents would try to persuade the intelligence agency demonstrating their quality as informers. The agency would estimate useless past secrets known by informer in order to compute how much the offering improvement could be rewarded in the future. The next illocution would help us represent such an argument:

$$reward\ (a,\ b,\ \varepsilon\ ,\ \psi\ ,\ t),$$

where *a* and *b* are agents, ε represents the desired improvements of the offering, ψ stands for the proofs of knowledge of other secrets revealed by agent *a* before time *t*. The privacy of the related secrets may be achieved by the secure computations mentioned above. And, finally, these proofs should be considered as a reward if agent *b* would accept ε .

Finally, we also propose an argument useful for introducing competence among intelligence agencies, as additional reward to the informers. An informer agent could appeal to the offering of another intelligence agency, demanding an improvement in the offering at least as good as the offering of the other agency. This argument can be represented by the following illocution:

$$appeal\ (a,\ b,\ \varepsilon\ ,\ \varphi\ ,\ t),$$

where *a* and *b* are agents, ε represents the desired improvements of the offering, φ stands for the offering received from another intelligence agency. An informer agent *a* would send at time instant *t* the offering from a third party to intelligence agency *b* in order to support the desired improvements contained in ε .

**Achieving anonymity**

Due to the secret nature of the information, informer agents will probably desire anonymity until an agreement is not yet reached, and while negotiation is still in progress.

Appealing anonymously to the offerings from other intelligence agencies entails certain relevant security problems:

1. The intelligence agency should be able to verify the authenticity and the integrity of such offering without asking directly the source of such offering.
2. Since possible improvements depend on the ownership of the offerings from other intelligence agencies, an informer should be able to show that such offerings belong to him/her, and no third party may eavesdrop them in order to use them inappropriately in other negotiations.
3. Besides hiding the real identity of the informer in each of the offerings, the proofs of ownership may not be linked. However, when the informer accepts the offering from an intelligence

agency, this agent should reveal the real identity of the informer showing a verifiable link between such identity and each of the proofs of ownership of offerings used during the negotiation.

The first of the requirements mentioned above can be easily satisfied through the digital signature ($SK_{ia}$) of the intelligence agency over the corresponding offering. These signed offerings might not be transferable if they include the identity public key of the informer ($PK_{id}$). However, such a solution will not satisfy the third requirement.

Therefore, we propose as an alternative to use of a different one-use key pair for each offering that has to be proved. The informer agent will generate locally such a pair of keys. The public-key ($PK_{off}$) will be included in the signed offering from the other intelligence agency. In this way, the informer agent will be able to use offerings from other intelligence agencies by showing the knowledge of the corresponding private key ($SK_{off}$) without revealing it. These processes are usually called *zero-knowledge proofs*. One party may show the knowledge of such private key, because it shows the ability to compute certain operations that strongly require the knowledge of the factorisation of that key pair. Then the offerings from the agencies can not be transferred, because even when the agreement is reached, the other party does not know the private key of the offering, and therefore, it will not be able to eavesdrop such offerings. The problem here is that the computational cost of generating, keeping and encrypting/decrypting with asymmetric ciphering may not be affordable.

A faster alternative is to use one-way hash functions to show afterwards the ownership of the offerings. Such offerings should include the hash function of a random seed (large enough). The one-way nature of such functions will make it very difficult to compute the reverse process. This alternative satisfies the third requirement, but once the agreement is reached, and, therefore, the random seed is shown to the intelligence agency in order to proof the ownership of such offering, that intelligence agency may freely eavesdrop such proofs in future negotiations. This circumstance hinders the satisfaction of the second requirement, so the offerings should include also a link between the real identity of the informer and the large random number used as input to the hash function.

Therefore, in the beginning of each negotiation, the informer agent will send a certificate linking the input of the one-way function with the real identity public key together with the computed hash output. It will take the following form:

$$\text{Hash}\ (\mathit{random\_input}),\ \{PK_{id},\ \mathit{random\_input}\}_{\text{encrypted with SKid}}$$

When negotiations succeed, such a link will be revealed when the real identity of the informer (Public Key of the informer) will be known and the authenticity of the offering may be verified but not eavesdropped (due to the inclusion of such certificate in the offering).

Figure 3 summarises the requirements that each alternative meets in a graphical way.

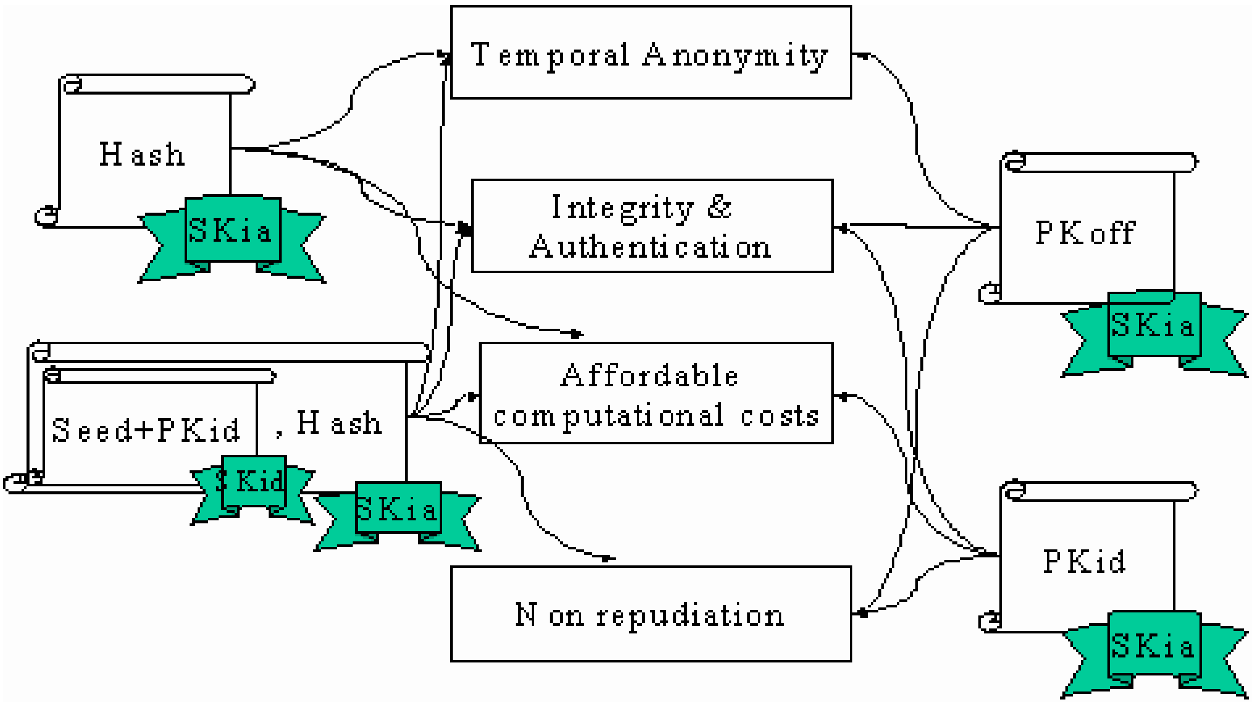



**Figure 3:** Overview of the proposed alternatives for achieving anonymity.

The inclusion of the identity public key (PKid) satisfies integrity, non-repudiation, authentication, and it may be affordable. The incorporation of an one-use-key pair for each offering (PKoff) satisfies anonymity but the implementation costs may become unaffordable. Nevertheless hash functions apparently solve this problem, but at the cost of sacrificing non-repudiation. Finally, our proposal satisfies all the requirements.

**Conclusions**

The negotiation protocol presented in this paper imposes few restrictions on its execution, and the space of negotiation is very open. Although it breaks the symmetry between the parties, informers would feel more comfortable if their own agents were leading the negotiation.

Intelligence agencies may have much more computational and storage capacity than informers. This circumstance may unbalance any negotiation, and therefore a certain level of anonymity is intended to diminish this disadvantage.

We have also studied how to prove certain knowledge without revealing the corresponding details, and how to preserve anonymity during negotiations when arguments such as past payoffs and offerings from other intelligence agencies are available.

Our proposal tries to make the negotiation dialogue more human through arguments commonly used in real-life negotiations; it also introduces competence among intelligent agencies and among informers. Further, our negotiation model intends to protect the interests of informers using secure computations, chains of hashes, etc. All these issues are tackled by our approach in a satisfactory manner.

**Notes:**


1. J. Von Neumann and O. Morgenstern, *The Theory of Games and Economic Behaviour* (Princeton University Press, 1944).

2. K. Binmore, *Fun and Games*. *A Text on Game Theory* (Lexington, MA: D.C. Heath, 1992).

3. J. Rosenchein and M. Genesereth, “Deals among rational agents,” in Proceedings of the International Joined Conference on Artificial Intelligence (1985): pp. 91-99.

4. K. Binmore and N. Vulkan, “Applying game theory to automated negotiation,” *Netnomics* 1-9 (Baltzer Science Publishers BV, 1999).

5. T.R. Gruber, “The role of common ontology in achieving sharable, reusable knowledge bases,” in *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning*, (Morgan Kauffman, 1991).

6. J. Rosenchein and G. Zlotkin, *Rules of Encounter: Designing conventions for automated negotiation among computers*, (MIT Press, 1994).

7. H. Raiffa, *The Art and Science of Negotiation*, (Cambridge, Mass.: Harvard University Press, 1982).

8. Sandholm and Lesser, “Issues in automated negotiation and electronic commerce,” in *Proceedings of the International Conference on MultiAgent Systems ICMAS’95*, (1995).

9. J. Broersen, M. Dastani and L. van der Torre, “Levelled commitment and trust in negotiation,” in *Proceedings of the Autonomous Agents 2000*, Workshop on Deception, Fraud and Trust in Agent Societies, (Barcelona, 2000).

10. R.G. Smith and R. David, “Frameworks for cooperation in distributed problem solving,” *IEEE Transactions on Systems, Man and Cybernetics* 11, 1 (1981): 61-70.

11. C. Sierra, N.R. Jennings, P. Noriega, S. Parsons, “A framework for argumentation-based negotiation,” *Intelligent Agents IV*, in *Lecture Notes in Artificial Intelligence*, no. 1365 (Springer, 1997), pp. 177-192,

12. Gasser, “Social conceptions of knowledge and action: DAI foundations and open systems semantics,” *Artificial Intelligence*, 47: 107-138.

13. S. Parsons and N.R. Jennings, “Negotiation through argumentation,” in *Proceedings of the Second International Conference On Multiagent Systems ICMAS’96*, (Kyoto, Japan, 1996): pp.267-274.

14. M. Karlins and H.I. Abelson, *Persuasion*, (London, UK: Crosby Lockwood, 1970).

15. Katya P. Sycara, “Persuasive argumentation in negotiation,” *Theory and Decision*, 28 (1990): 203-242.

16. Sierra, Jennings, Noriega, and Parsons, “A framework for argumentation-based negotiation.”

17. Kraus, “Reaching agreements through argumentation: A logical model,” in *DAI Workshop’93*, (1993): pp.233-247.

18. ISO 7498-2 1989 *ISO/IEC 7498 Security Architecture, part 2.* Open System Interconnection Reference Model. (1989).

19. B. Schneider, *Applied Cryptography: Protocols, Algorithms and Source Code in C,* 2nd Ed., (New York: John Wiley and Sons, 1996).

20. R. Rivest and A. Shamir, “Payword and Micromint: two simple micro-payment schemes,” in *Proceedings of 1996 International Workshop on Security Protocols*, ed. Mark Lomas, *Lecture Notes in Computer Science* no. 1189, (Springer, 1997), pp.69-87.

21. J. Searle, *Speech Acts*, (Cambridge University Press, 1969).

22. R. Neches, R.S. Patil, R.E. Fikes, P.F. Patel-Schneider, D. McKay, T. Finin, and T. Gruber, “DARPA Knowledge Sharing Effort,” in *Proceedings of the Annual International Conference on Knowledge Acquisition*, (Cambridge, MA, October, 1992).

23. A. Yao, “Protocols for secure computations,” in *Proceedings of the 23rd IEEE Symposium on Foundations of Computer Science*, (1982): pp. 80-91.

24. J. Carbo, J.M. Molina, and J. Davila, “Privacy of Trust in Similarity Estimation through Secure Computations,” in *Proceedings of the International Workshop on Trust and Privacy in Digital Business, 13th International Conference on Database and Expert Systems Applications*, (Aix-en-Provence, 2002).

---

**JAVIER CARBO** is currently lecturer at the Computer Science Department of the Carlos III University of Madrid (SPAIN). He is with the Laboratory of Adaptive Complex Systems at the department. He has previously worked at the Artificial Intelligence Laboratory of the University of Savoie (FRANCE). He is currently finalising a Ph.D. on reputation and trust issues of agent-mediated electronic commerce. He obtained a Computer Science degree from the Politecnica University of Madrid in 1997. Mr. Carbo has published over 15 papers. He acts as a reviewer of IASTED international conferences on Artificial Intelligence. He has also acted as a session chair of the IEEE International Conference on Systems, Man and Cybernetics. He took part in an U.N. funded research project, two ESPRIT programs and other national projects. His interests are in automated negotiations, multi-agent systems, electronic payments and fuzzy logic. E-mail: jcarbo@inf.uc3m.es, Phone +34 916249105, Despacho 21B12, Depto. Informatica, Univ. Carlos III, Av. Universidad 30, Leganes 28911 Madrid (SPAIN)

**JOSÉ M. MOLINA** received a degree in Telecommunication Engineering from the Universidad Politecnica de Madrid in 1993 and a Ph.D. from the same university in 1997. He has worked in the Signal Processing and Simulation Group of the same university since 1992, participating in several national and European projects related to Radar Processing. In 1993 he also joined the Computer Science Department of the University Carlos III of Madrid, being enrolled in the Systems, Complex and Adaptive Laboratory performing research on soft computing techniques (NN, Evolutionary Computation, Fuzzy Logic and Multiagent Systems). He is author of more than 10 journal papers and 70 conference papers. His current research focuses on the application of soft computing techniques to security in e-commerce, information retrieval from web, problem solving in web domains, radar processing, air traffic control, etc. E-mail: molina@ia.uc3m.es, Phone +34 916249116, Despacho 21B15, Depto. Informatica, Univ. Carlos III, Av. Universidad 30, Leganes 28911 Madrid (SPAIN).

**JORGE DAVILA** received a degree in Chemistry from the Universidad Complutense de Madrid and a Ph.D. from the same university. In 1991 he joined the Computer Science Faculty of the Universidad Politecnica de Madrid, where he is teaching and researching in Cryptography and Information Security since then. From 1993 he leads the Cryptology Laboratory of the Faculty. E-mail: jdavila@fi.upm.es, Ph. +34 913366934, DLSIIS, Facultad de Informatica, Univ.

Politécnica, Campus de Montegancedo, Boadilla del Monte 28660 Madrid (SPAIN).

**BACK TO TOP**

---


**Information & Security. An International Journal**
**e-mail: infosec@mbox.digsys.bg**

# Argumentative Negotiations with Anonymous Informer Agents

*Javier Carbo, José M. Molina and Jorge Davila*

**Keywords:** Multi-agent system, argumentative negotiation, intelligence agencies, informers, anonymity, security

**Abstract:** In this paper we present a scheme of multi-agent argumentative negotiation in the generic domain encompassing intelligence agencies and informers. Two agents, an intelligence agency and a potential informer, negotiate multiple criteria through the sequential exchange of agreement offers and counteroffers. The goal of the negotiation phase in our work is to persuade the other party by argumentative reasons. We have studied several typical arguments, which may be useful in this context, such as ultimatums, promises of future fidelity, past behaviour, and other offerings received. Furthermore, privacy protection is one of the major issues of such negotiation of secrets with informers. This can be accomplished through the exchange of arguments in counteroffer attributes and by using encryption techniques.

**full text**

Authors: **David Camacho, Jose M.Molina, Daniel Borrajo, Ricardo Aler**
Title: **MAPWEB: Cooperation between Planning Agents and Web Agents**
Year of issuance: **2002**
Issue: **Information & Security. Volume 8, Number 2, 2002, pages 209-238**
Hard copy: **ISSN 1311-1493**

---

# MAPWEB: COOPERATION BETWEEN PLANNING AGENTS AND WEB AGENTS

David CAMACHO, José MOLINA, Daniel BORRAJO and Ricardo ALER

---

**Table Of Contents:**



---

## 1. Introduction

Nowadays, there is a vast (and growing) amount of information stored in the Web available for any user connected to the network. This information is heterogeneous and distributed. Web information could be used by the users to solve many different problems if only they could spend enough time searching, retrieving, and analyzing the data. Internet provides a lot of Web applications like search engines and meta-search engines that enable the users to look for the information they need. However, currently it is impractical to build a single and unified system that combines all possible information sources that could be useful to the users. Some of the reasons are summarized below:

- The number of information sources in the web grows exponentially.

- There are a lot of Web information sources that provide similar information, each one using its own representation of the information. For instance, a traveler might want to find suitable plane companies to get to his/her destination. However, different companies will display on the web similar data but using different representations.

- Different information sources usually provide different kinds of information and it is not always easy to combine them to achieve common goals. For instance, in Non-combatant Evacuation Operations (NEO) domains, it would be useful to combine the information coming from weather forecast sites with the information obtained from Geographical Information System (GIS) servers.

- The information stored could change dynamically over time. A short-term weather forecast site is a good example.

Due to the previously elaborated problems, current Web search engines basically rely only on purely syntactical textual information retrieval. There are only a few approaches that try to integrate a set of different and specialized sources, but unfortunately it is very difficult to develop and to maintain this kind of systems.[1] Therefore, users cannot use heterogeneous information to obtain satisfactory results in problem solving in a short time and with high quality. It is true that there are many systems that extract, filter and represent efficiently the information obtained from the Web. However, most of these systems are focused mainly on the amount of information retrieved.[2]

Integrating heterogeneous information is one of the main goals of MAPWeb. However, having complete and high-quality information is not necessarily an end in itself. If the user wants to solve complex problems using that information, the system must include intelligent elements able to reason in complex domains. For instance, a traveler needs to be provided with good plans that combine different means of transportation in an efficient manner. Similarly, NEO and military operations need intelligent systems to move units and supplies on the terrain in a coordinated and efficient way. Artificial Intelligence (AI) provides software components that fill in that gap: planning systems that find good quality plans in complex domains, machine learning systems that learn from experience in these domains, etc. In our work, we apply such AI techniques but in a Multi-Agent System (mas) framework (a part of what is called Distributed Artificial Intelligence (dai)). These systems are built using a set of modular components, or *agents*, that are **specialized** in solving a particular aspect of a problem.[3] This decomposition allows each agent to use the most appropriate paradigm for solving its particular problem.[4] Every mas uses the agent concept, which is extensively described in several publications.[5] The main characteristics of a mas can be summarized as follows:

- Each agent has an incomplete amount of information or does not have the required abilities to solve the entire problem.

- There is no centralized control.

- Data is not centralized; therefore agents must share their data.

- System execution is asynchronous; an agent can be working and receiving queries simultaneously.

- Each agent has an internal state. It is also able to reason about the environment and possibly learn from experience in order to improve its behavior.

In our work, the agent-based framework provides certain benefits, namely:

- First, multi-agent systems are societies of (usually) heterogeneous software components, using, however, in their communication a common language. Therefore, mas address directly the problem of integrating heterogeneous systems, each one handling different kinds of information. This is the problem that complex web information retrieval systems have to face, as has been mentioned before.

- mas are often used to solve AI problems, such as planning, scheduling, learning, and they have shown their value, due to the fact that:

  ⇒ Different agents can combine their abilities in a **synergetic** manner. This has been clearly shown in the so called multi-strategy learning systems where different systems provide different characteristics useful to achieve a common goal.[6] However, this is not the only example. For instance, it has been shown that different planners work well in different domains.[7] Therefore, in some cases it would be a good idea to combine different planner agents in the same mas system.[8]

  ⇒ They offer modularity, flexibility, and adaptability. A mas uses a common language and, thus, provides a means for communication between heterogeneous agents. Hence, it is easy to add new agents with new abilities, if required. These characteristics are essential in complex, large or unpredictable domains.[9]

  ⇒ mas are inherently parallel, in this way facilitating the efficient execution of the computationally complex problems associated with AI.

However, integrating and coordinating different agents is a complex problem in itself that has to be addressed.[10,11] When interdependent problems arise, the agents in the system have to cooperate in order to ensure that the interdependencies are properly handled.

This paper presents a distributed multi-agent architecture – MAPWeb – that accepts queries from the users. These queries are actually the problems that have to be solved. As a result the system produces possible solution schemata by means of AI problem-solving techniques (planning and learning), which are then validated and completed using the information available on the Web.

This paper is divided into seven sections: Section 2 presents a review of the related work in Multi-agent systems; Section 3 describes the MAPWeb architecture; Section 4 analyzes how the system interacts with the user and finds solutions; Section 5 presents an example application domain of the designed system; Section 6 summarizes the conclusions; and finally, Section 7 shows the future lines of work.

## 2. Related Work

There are several approaches that attempt to work with the information stored on the Web. These approaches focus mainly on the process of retrieval of (usually textual) information, but only few of them try to reason with that information. This section analyzes these systems and illustrates how they handle the stored data. We will focus on the *agent-based* systems (like *Web and Intelligent agents*) and the *multi-agent-based* systems that have been developed and deployed recently.

Web and Internet applications can be classified in different ways. The following classification focuses mainly on how these systems use the available data:

1. **Simple Web-Applications:** Systems that search, retrieve and store information, like *searchers, meta-searchers* or any other popular information retrieval applications. The main goals of these systems are the search and retrieval of Web information.

2. **Complex Web-Applications:** Systems that transform the obtained information, share with other systems its knowledge, and even could cooperate with other systems to obtain *solutions* that may be useful to the users. These kinds of systems have a wide range of characteristics that attempt to achieve more complex tasks than just retrieving information.

Figure 1 displays a possible classification of the most common Web applications. Solid lines show that the majority of a given kind of system belongs to the designated class and discontinuous lines show that some applications could be built using only a subset of the characteristics so that intelligence and/or robustness of the systems is increased.

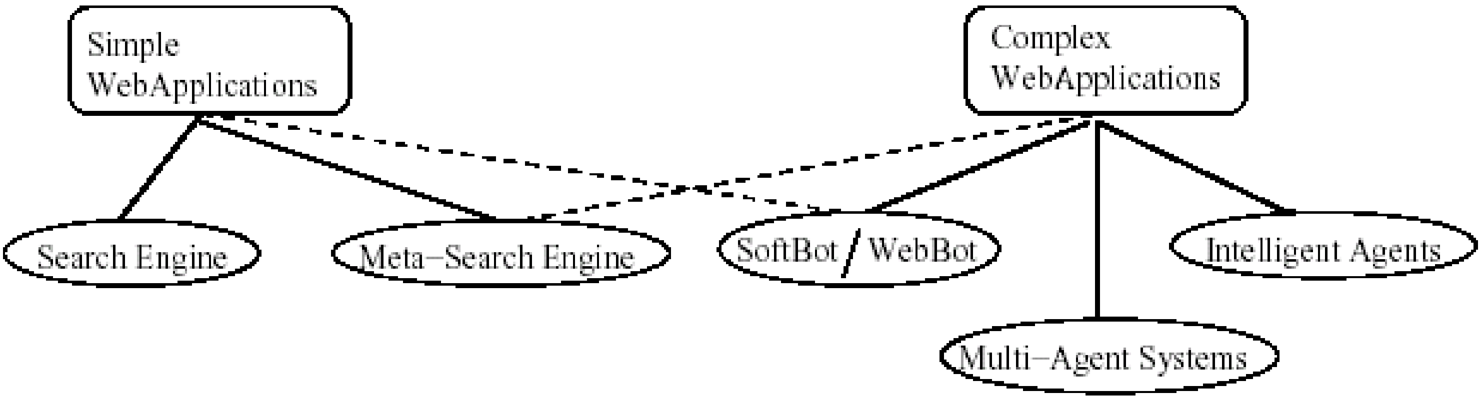



**Figure 1:** Generic classification of Web systems.

### *2.1. Intelligent Agents*

Intelligent Agents are software entities that assist people and act on their behalf. They make the user's life easier, save his/her time, and provide a simplified view of a complex world. Any *Intelligent Agent* tries to assist, advise or learn from past experience or from other agents' experience to anticipate the requirements of the user. In fact, the agent-based technology is a combination of various technologies including, but not limited to, neural networks, expert systems, fuzzy logic, machine learning, planning.[12]

It is a difficult task to characterize accurately what is an agent. There is extensive literature on the topic.[13,14] The following features can be used to characterize an intelligent agent:

- Agents are *pro-active* in nature (although they can and will be reactive as well).[15]
- Agents can *learn* as a result of their action – not to mention from their mistakes.[16,17]
- Agents can be *predictive* in nature.[18, 19]

- Other key attributes of the paradigm include *autonomy*, *security*, *personality*, and *mobility*.[20], [21] However, an agent need not possess all these characteristics. The fact that an agent can move from one environment to another is not a requirement in all cases.

- Lastly, agents are *social* in nature. They can collaborate with other agents as well as delegate tasks to subordinate or "better suited for the job" agents.[22]

Different and successful intelligent agents have been developed recently. In the following, some of these agents will be briefly described:

- **`Softbot.`** This agent interacts with a software environment by using and interpreting the environment feedback. The softbot effectors are unix commands that enable the agent to change the state of the user environment.[23]

- **`SodaBot`** is a general-purpose software agent user environment and construction system. Its main component is the *basic software agent* that is a computational framework for building agents; it is essentially an *agent operating system*. Through the definition of a new programming language it is possible for the users to implement a wide-range of typical software agent applications, such as on-line assistants or meeting-scheduling agents.[24]

- **`SIMS`** and **`Ariadne.`** These intelligent information agents are focused mainly on information retrieval and integration of different kind of information sources. `SIMS` focuses on the integration of well-structured databases,[25] while the `Ariadne` project deals with accessing information from more loosely structured Web sources.[26]

### *2.2. WebAgents*

Currently there is an enormous number of Web applications that offer different services to Internet users, such as search and meta-search engines (*Lycos*, *Altavista*, *Yahoo*), e-commerce markets, auctions, web directories, etc.[27] As we have said in Section 1, due to the current evolution of the Web (and other on-line information sources), it has become a necessity to provide some sort of intelligent assistance to the users. WebAgents are applications that are able to consult the best Internet sites and perform agent specific tasks, such as retrieving, processing, tracking and submitting required information. WebAgents perform specific Internet tasks. From this point of view, the functionality of WebAgents is given by the agents installed on the system and their specific purpose. There is a lot of research and development on this kind of systems. Here we only briefly mention some of the developments:

- **`MetaCrawler.`** The MetaCrawler Softbot is a parallel Web search service that provides unified interface by which any user can query popular general-purpose Web search engines, such as *Lycos* or *Altavista*. MetaCrawler has some characteristics that enables it to obtain results of higher quality than simply showing the output from the search service.[28]

- **`Letizia`** is a *user interface agent* that assists users browsing the World Wide Web. While the user operates a conventional Web browser, the agent tracks user behavior and attempts to anticipate items of interest by doing concurrent, autonomous exploration of links from the user's current position. The agent automates a browsing strategy consisting of best-first search augmented by heuristics derived from inferring user interest from its browsing behavior.[29]

- **`WebWatcher`** is a "tour guide" agent for the World Wide Web. Once a user enters to

WebWatcher what kind of information he/she looks for, it accompanies the user from page to page. While the user browses the web, it highlights hyper-links that it believes could be of interest. Its strategy to give advice is learned from the feedback from earlier tours. Currently, WebWatcher is online only on an irregular basis.[30]

- **WebPlan.** This intelligent Web agent has been developed at Kaiserslautern Universtity. WebPlan is a search assistant for domain-specific search on the Internet based on dynamic planning and plan execution techniques.[31]

### *2.3. Multi-Agent Systems*

Due to the growing importance of agent-based technologies in the development of software systems, there are several commercial and research agent development toolkits. It is very difficult to select an appropriate toolkit, as each toolkit has been designed for a certain *architecture* or *paradigm*. We will only examine several popular toolkits and deployed mas.

- **AgentBuilder.** This is a very popular commercial toolkit for building and testing agent-based software. Agents constructed using AgentBuilder communicate using kqml (Knowledge Query and Manipulation Language).[32] It makes it possible to develop and extend the standard kqml performatives (or messages) to include additional performatives.[33]

- **JAFMAS.** This toolkit provides a framework that helps developers to structure their ideas into specific agent applications. It directs the development from a speech-act perspective and supports multicast and directed communication, kqml or other speech-act performatives. It also performs some analysis on the multi-agent system coherency and consistency.[34]

- **JADE** (Java Agent Development Framework) is a software development framework aimed at developing multi-agent systems and applications, conforming to fipa standard for intelligent agents.[35] JADE can be considered as an *agent middle-ware* that implements an *Agent Platform* and a development framework.[36]

- **JATLite** is a framework for creating multi-agent systems. JATLite includes a message router (*agent message router* or simply *AMR* agent) that supports message buffering, allowing agents to fail and recover. Agents can send and receive messages using kqml. Message buffering also supports a name-and-password mechanism that enables agents to move freely between hosts.[37]

- **Kasbah** is a virtual market place on the Web where users can create autonomous agents that buy and sell goods on their behalf. Users can specify parameters to guide and constrain the agent's overall behavior. Any intelligent agent in Kasbah is an object (an instance of a class) and the market place allows the user to create buying and selling agents, which then interact in the market with other agents. The agents themselves are not very smart, although they are completely autonomous. Agents do not use ai or Machine Learning techniques. The interesting aspect of Kasbah is its multi-agency. It is a good framework for testing different important characteristics of this kind of systems, such as *negotiation*.[38]

- **MPA.** The Multiagent Planning Architecture is a framework for integrating diverse technologies into a system capable of solving complex planning problems. MPA has been designed for application to planning problems that cannot be solved by *individual systems*, but rather require the coordinated effort of a diverse set of technologies and human experts.[39]

- **CMUexpress** is a mas architecture developed at Carnegie Mellon University (CMU). Its purpose is to plan, execute plans, and monitor its performance. It has been applied to Non-combatant Evacuation Operations (NEO). In this specific case, the entire system integrates about twenty agents. In particular, it includes MMM (a user interface developed at Stanford Research Institute- SRI), Ariadne (described above), and the already mentioned CMUExpress. The goal is to locate, pick up, and carry civilians to a safe place. The agents collaborate in the following manner. First, Ariadne locates the civilians. Then, CMUExpress provides routing plans to transport them, in addition to monitoring the on-going plan and reacting to events. CMUExpress can use the tracking information provided by Ariadne, that is obtained from an on-line web-site.[40]

- Finally, there is a hierarchical multi-agent system developed at DERA (UK) to plan military activities (i.e., moving troops on a terrain) and execute them. Its aim is to combine deliberative and reactive behavior. The agents in the society are organized in a hierarchical military manner. For instance, there is a Squadron Commander agent, a Troop Commander agent, a Tank agent, etc. This framework enables the more reactive behaviors of the agents at lower levels of the hierarchy to be guided by the more deliberative planning of the agents above them in the hierarchy. In particular, a constraint planner (deliberative) and an anytime planner (reactive) are combined within the hierarchy.[41]

## 3. MAPWeb: A Multi-Agent Architecture for Reasoning on the Web

As already mentioned, the main advantage of using mas techniques is the flexibility and adaptability of the resulting system. A mas could consist of several heterogeneous elements. These elements, or *agents*, can play different programmed roles, could execute different functions, and could modify their behavior dynamically.

MAPWeb is a mas approach that integrates heterogeneous agents. These agents assemble a set of *"logic-layers"* between the users and the Web. The architecture *hides* the Web from the users. This facilitates the user in coping with the overload of information. Figure 2 illustrates the four-layered architecture of MAPWeb.

1. **Physical World:** it represents the users.

2. **Reasoning Layer:** this layer connects any physical agent (usually human) with a set of systems that allows the agents to access the desired information.

3. **Accessing Information Layer:** this layer retrieves the information from distributed sources (like the Web) and represents it in an understandable fashion to the previous layer.

4. **Information World:** it represents all the information available on networks, computers, or any other kind of electronic support. This *"world"* is accessible only through information retrieval systems.

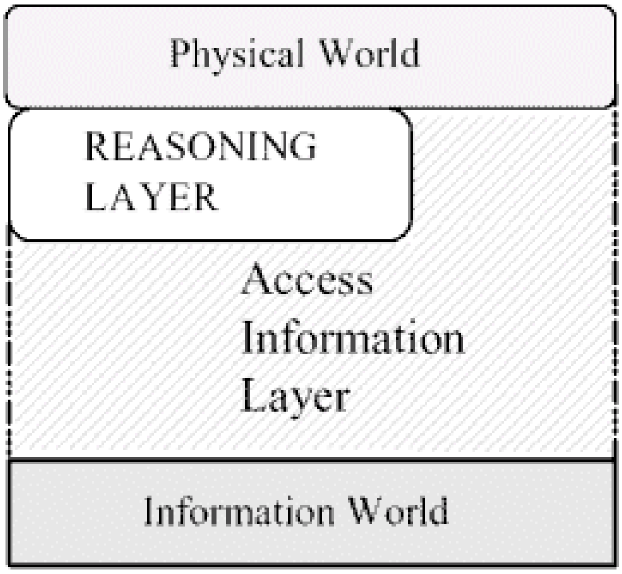



**Figure 2:** World/Web Layers.

How MAPWeb implements the above described multi-layer architecture, can be seen in Figure 3. The system is composed of a set of agents that can communicate, share knowledge and cooperate in order to find solutions to the problems posed by users.

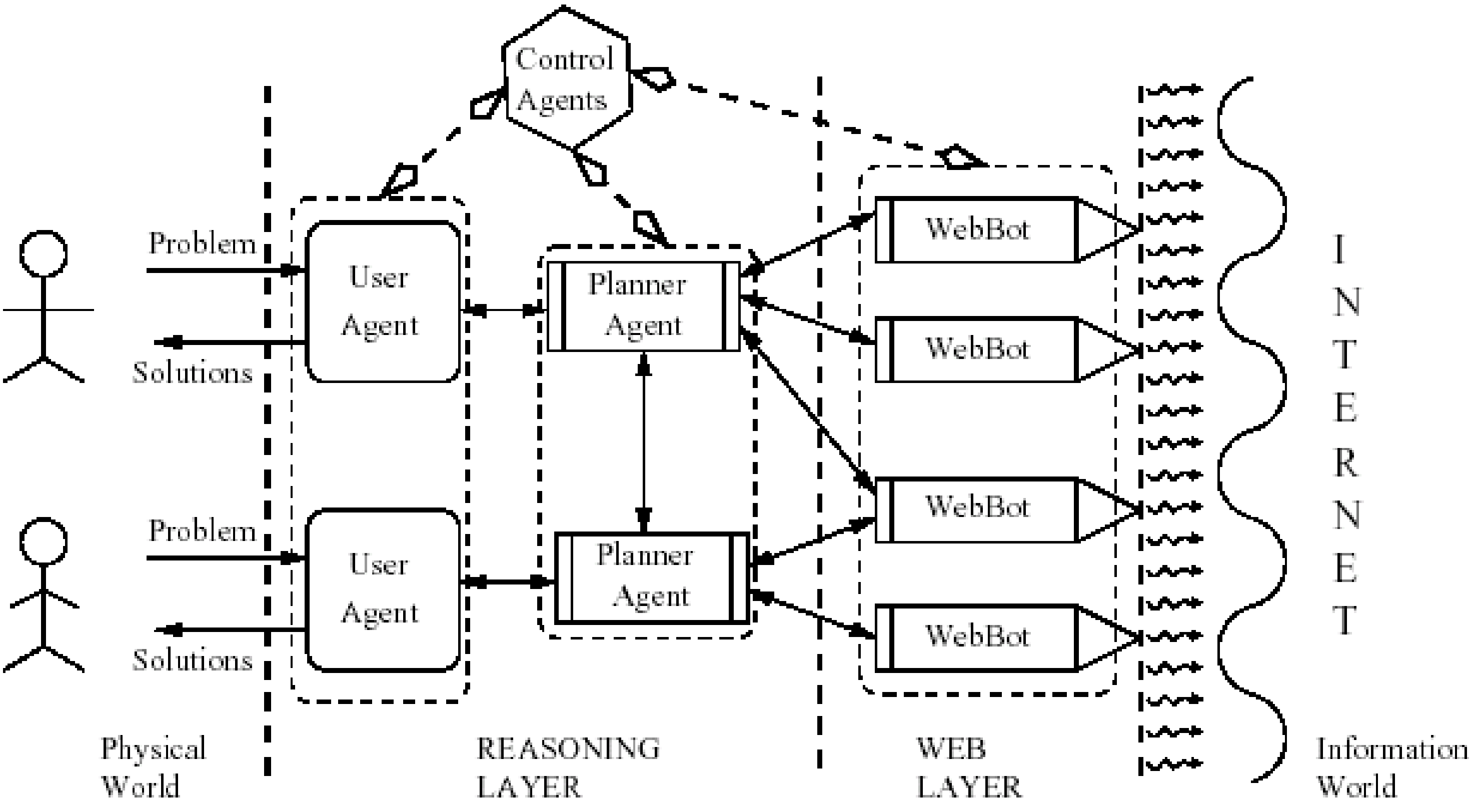



**Figure 3:** MAPWeb general Architecture.

This architecture has been designed to deal with some frequent problems existing on the Web. In order to accomplish this, it is necessary to use an internal knowledge representation shared by the agents, and different reasoning techniques that enable the agents to look for new solutions. MAPWeb is a mas approach that integrates different heterogeneous agents with diverse roles into the agent society. The types of agents used can be summarized into the following categories:

- **UserAgent:** this agent connects the *physical world* with the *reasoning layer*. It takes user queries and displays to the user the solution(s) found by the system. UserAgents capture problem queries from the users and send them further to a reasoner-agent. PlannerAgent is the only currently developed reasoner-agent, but various kinds of reasoner-agents, such as LearningAgents, will be developed in the future. Afterwards, the reasoner-agents are responsible for finding solutions to the problem.

- **ControlAgents:** These agents belong to the *reasoning layer* and considering the organizational structure of the system, there are two different types of control-agents in MAPWeb: ***ManagerAgent*** and ***CoachAgents***. Their main roles are summarized below:

  ⇒ **ManagerAgent:** It directs the insertion and deletion of agents from the system. This agent is responsible for building dynamic *teams* of agents specialized in different problem solving activities.

  ⇒ **CoachAgent:** This agent controls a set of heterogeneous agents that represent a *team*, which accepts problems from any agent (software or human) in the system and attempts to solve them.

  Figure 4 illustrates the relationship between these kinds of agents and the rest of the agents in MAPWeb. Agents are organized in teams, each one is managed by a coach. The whole system is leaded by a manager. Each UserAgent, PlannerAgent or WebAgent might belong to several teams if necessary for the proper work of the team.

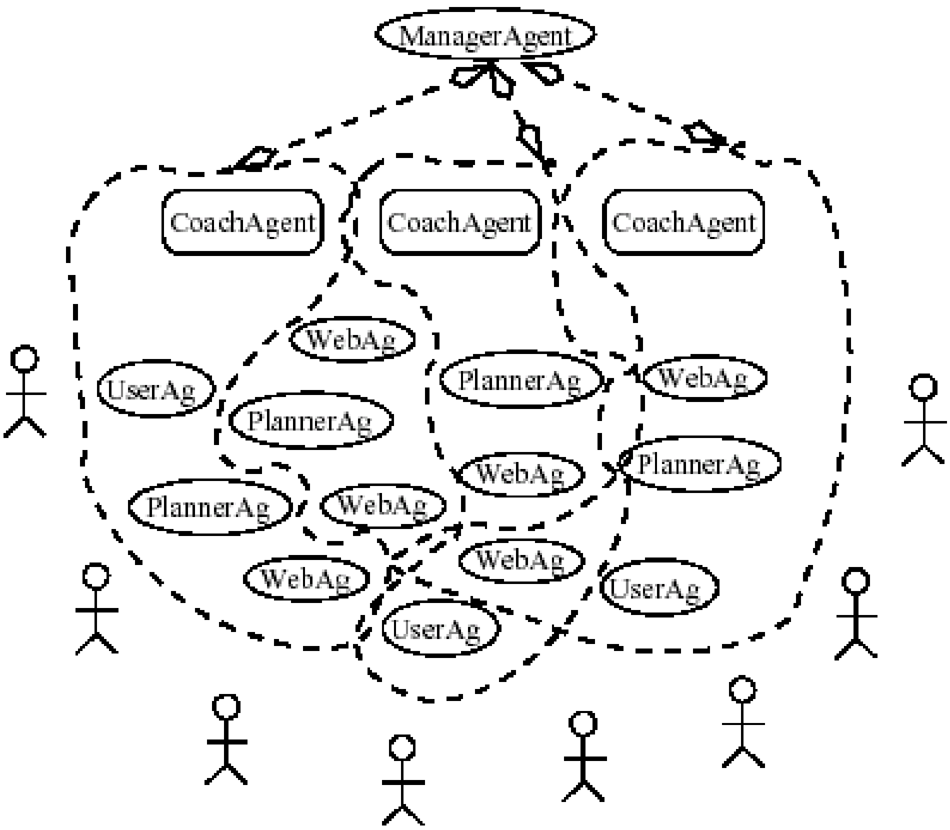


**Figure 4:** Manager and Coach Agents Organization.

- **PlannerAgent:** This agent (belonging to the *reasoning layer*) receives a planning problem, builds an abstract representation of it, and solves it. PlannerAgents have different abilities, such as communication and planning.

- **WebAgent:** These agents belong to the *accessing information layer* and connect the *reasoning layer* with the *information world*. Its main goal is to complete the details of the abstract plans obtained by the PlannerAgents. It receives that information from the Web.

Some of the underlying modules (see Figure 5) of any MAPWeb-agent are:[42]

1. *Control module:* it manages all possible tasks performed by the agents. This module is basically made of an agenda, some policies, and a set of specialized skills.

2. *Knowledge module:* this module is used by the different agents to store their own knowledge.

3. *Skills module:* this module implements the specialized skills of any agent in the system.

4. *Communication module:* it implements the communication protocol with other system agents (UserAgents, PlannerAgents, CoachAgents, or WebAgents). This module is implemented using two sub-modules:

    - *Transport module:* it implements a TCP/IP network-level communication between two agents running on different computers.

    - *Language module:* it implements a standard version of kqml [43] that makes it possible to use a common language between two agents in MAPWeb.

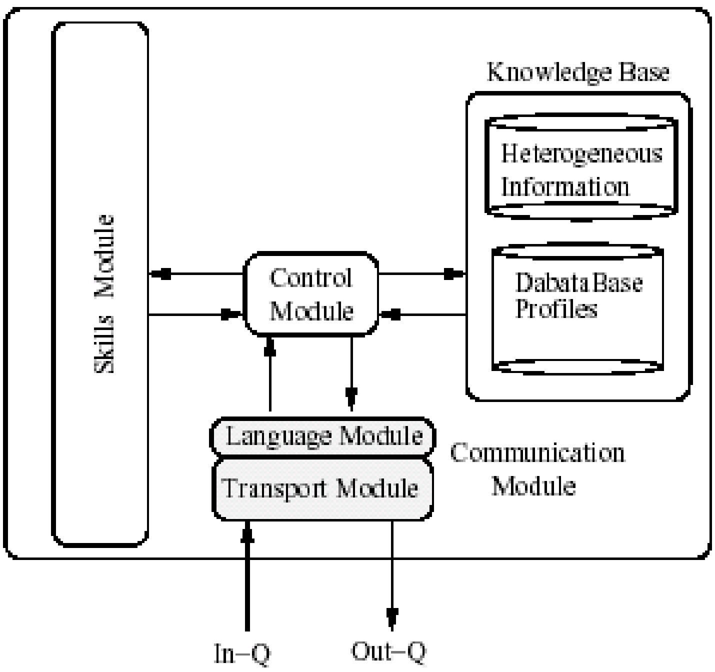



**Figure 5:** Skeleton-Agent in MAPWeb.

The following subsections give a more detailed description of the different agents: roles, architectures, and organization.

### 3.1. UserAgents

The main role of UserAgents is to connect the users with MAPWeb. Each UserAgent uses a set of Graphical User Interfaces (gui) to communicate with the users and an implementation of the standard language kqml to communicate with other agents in the system. Figure 6 presents a modular description of the UserAgent architecture.

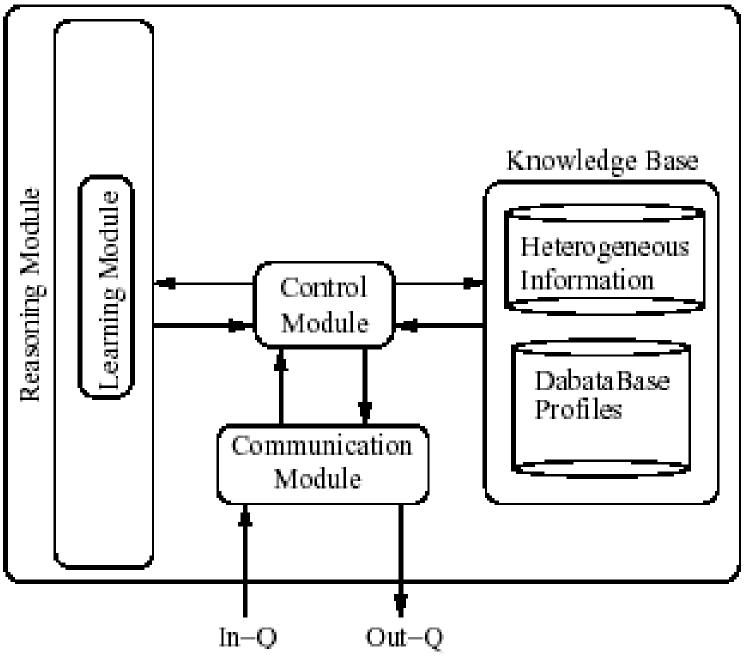



**Figure 6:** UserAgent Architecture.

The *Knowledge Module* is used by the UserAgent to store a set of different user profiles and successful old solutions, that can be used by UserAgent (applying its learning skills in the *Learning Module*) to analyze and customize the system.[44]

The main goals of a UserAgent are:

- To accept problems from users and to present the solutions found by MAPWeb.
- To analyze the problems and to obtain homogeneous representation for them.
- To communicate with PlannerAgents in order to request solutions.

For the accomplishment of the previously described goals, it is necessary to provide, for each particular domain, the specific set of guis that can represent all the necessary input/output information for communication with the external world, and to define an ontology that allows the other agents in the system to know the type of problem that has to be solved. Section 5 will present the set of guis for a considered domain.

### 3.2. PlannerAgents

Any PlannerAgent has a modular architecture where each module has its own capabilities and tasks. These are the reasoning agents in the system. Figure 7 depicts the PlannerAgent's modular representation.

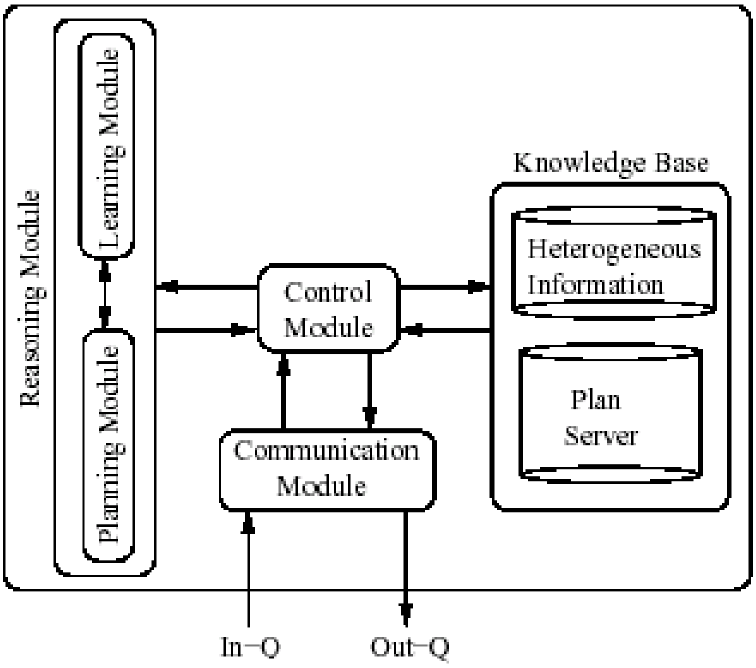



**Figure 7:** PlannerAgent Architecture.

Some of the most interesting characteristics of PlannerAgent are:

- *Communication module:* it implements a subset of specific *performatives* (speech-acts in kqml) used by PlannerAgents to share plans or sub-plans.

- *Knowledge module:* Stores useful information for the agents. It is composed of two main sub-modules:

  ⇒ *Heterogeneous Information:* This sub-module stores useful data (heterogeneous information) about the application domain, planning operators, heuristics, information about other agent characteristics, statistics information, etc.

  ⇒ *Plan server:* This module stores old plans or sub-plans that can be used in finding new solutions.[45]

- *Control module:* it is used to manage the various agent modules. Some of its main functions are:

  ⇒ To handle abstract solutions; they should be validated using the information acquired from other agents, or from other heterogeneous information sources.

  ⇒ To build an agenda that handles its own tasks and the questions posed by other agents.

  ⇒ To deal with all possible answers given to questions asked by other agents and/or users.

- *Reasoning module:* It is mainly comprised of two sub-modules:

  ⇒ *Learning modules:* They can modify the system behavior if the obtained solutions are successful in solving user problems. Currently, a Case-Base Planning Module is being developed, and it is used to gain efficiency in the planning process by retrieving and adapting past stored solutions; it avoids performing the planning process.[46]

  ⇒ *Planning module:* it performs the actions necessary to solve the user problem. Currently, the planning module uses the non-linear planner prodigy4.0.[47]

The PlannerAgents use a planner as main reasoning module. The agent generates an abstract representation of the problem and the specific user queries (given by the UserAgent). Then, it uses a planner to obtain a very abstract solution (or solutions) of the problem, and finally cooperates with the WebAgents to fill in the details of these abstract solutions.

### 3.3. ControlAgents

As previously described, there are two different types of Control agents in MAPWeb. They have identical architecture (see Figure 8) but different roles.

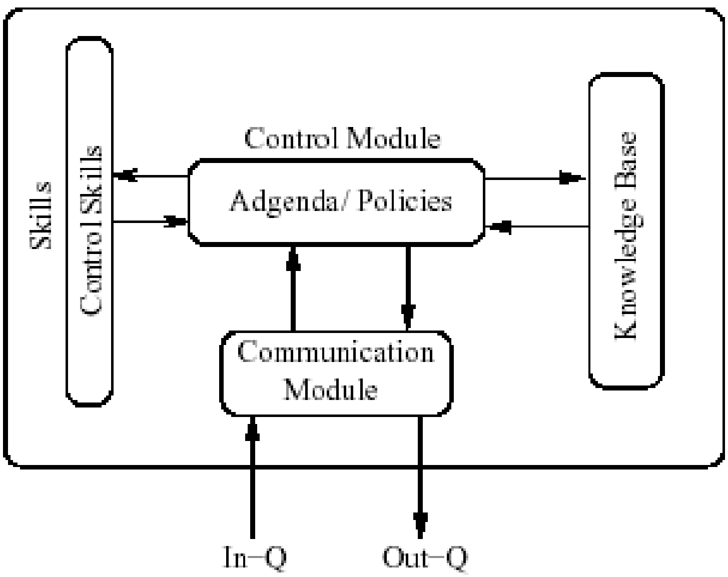



**Figure 8:** Generic ControlAgent Architecture.

We could outline the differences as follows:

1. ManagerAgent:

- There is only one ManagerAgent.
- It is responsible to add and remove other agents from the system.

- It controls which agents are active in the agent society.
- It groups agents in teams.
- It determines which are the agents shared by the different teams.

2. CoachAgent:

- It controls a team of agents, guaranteeing stability and smooth operation of the active agents.
- It reports problems to the ManagerAgent. For instance, when a new agent is required for the team.
- It guarantees that the agendas of the team members are coherent.

To function correctly, MAPWeb (for any possible multi-agent topology) needs at least one Manager and one Coach agents to build teams that will be able to reason about the user problems.

### *3.4. WebAgent*

The *WebAgents*, like the other system agents, have their own modular architecture (it is shown in Figure 9). A WebAgent handles (control module) the questions received from other agents (PlannerAgents), and translates these questions into queries to the Web (Internet access module). The answers from the Web will be filtered and stored in a data base (database from web). This useful information will be sent later to the PlannerAgent. WebAgents know various places where to look for the requested information.

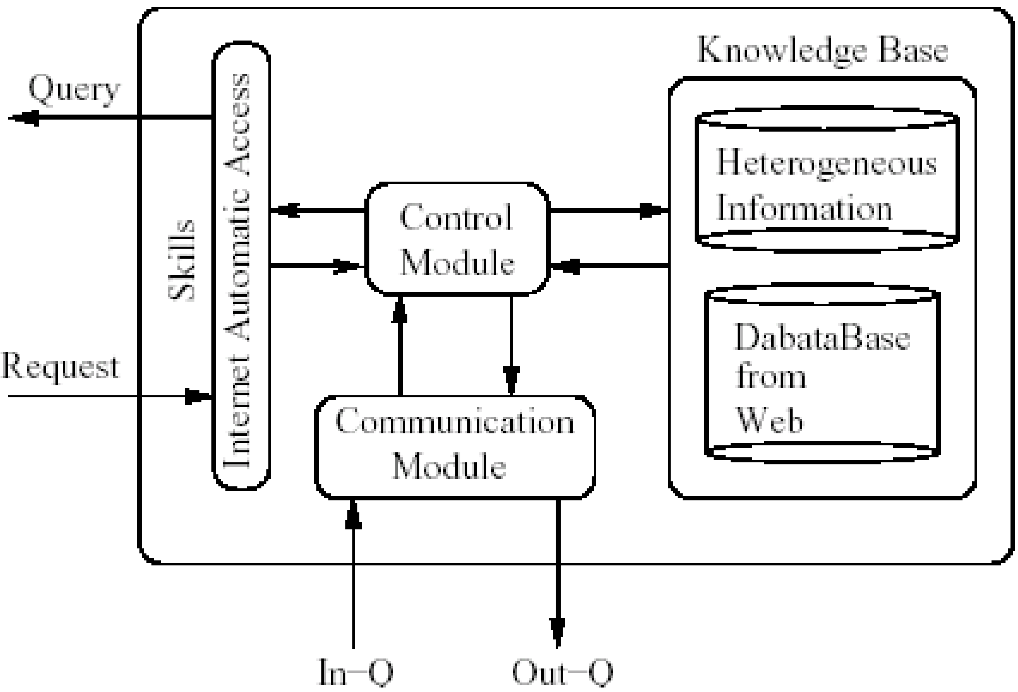



**Figure 9:** WebAgent Architecture.

Although MAPWeb has a very general architecture and it is possible to apply MAPWeb to different domains, the paper presents an implementation of a set of WebAgents specialized in the task of retrieving, filtering, and representing the necessary information from the Web for a specific domain (see Section 5).

## 4. Problem Solving and Cooperation in MAPWeb

MAPWeb has an architecture where different agents have to cooperate in order to reach a solution. Different agents need to share their knowledge and skills to complete the abstract solutions obtained by the PlannerAgent. MAPWeb's success depends on the following factors: sharing knowledge to obtain new solutions and using different Web and reasoning skills by the MAPWeb agents to find useful solutions for the users. In what follows the format for sharing and communicating knowledge, and the generic cooperative-solving process in MAPWeb, are analyzed.

### *4.1. Sharing Information between Agents*

Agents in MAPWeb use a common representation for the knowledge. This characteristic facilitates the process of sharing and reasoning with the knowledge. Agents use **performatives** in their communication. Any performative contains an implicit order to another agent. For communication between system agents, a subset of the kqml format is currently being used.[48] This format is shown in Table 1.

**Table 1:** Some performatives in MAPWeb.

| Performative | Format |
|---|---|
| *achieve* | (:content (FLY Company MAD ZAZ…)<br>:language JAVA<br>:ontology Electronic-Tourism<br>:in-reply-to MAPWeb<br>:sender PAgent1<br>:receiver WBot1) |
| *tell* | (:content (FLY IBERIA 323 Price…)<br>:language JAVA<br>:ontology Electronic-Tourism<br>:in-reply-to MAPWeb<br>:sender WBot1<br>:receiver PAgent1) |

This example illustrates the representation of two performatives: achieve and tell. The first performative (achieve) is sent by a PlannerAgent (*PAgent1*) to a WebAgent (*WBot1*) asking for Web information the WebAgent is specialized in. The second performative (tell) is the reply from the WebAgent to the PlannerAgent; it stores the information retrieved by the agent *WBot1*.

There exist other kqml performatives implemented by MAPWeb agents to manage the group of agents and to allow agent negotiation, such as accept, reject, register, unregister, delete, insert.

### *4.2. Cooperation in MAPWeb*

This section describes how UserAgents, PlannerAgents, and WebAgents cooperate to solve problems. From a generic point of view, a problem is a pair (***initial situation***, ***final situation***). An example of a problem is the following: a person intends to fix (final situation) a broken car (initial situation). A solution to a problem is the sequence of actions to be performed so as to get from the initial to the final situation (called a `plan`). Usually, actions are defined in terms of `operators`. For instance, `screw(x)` could be an operator denoting the use of the screwdriver on any screw `x`. Therefore, a solution to the car fixing problem could be anything like the plan presented in Figure 10.

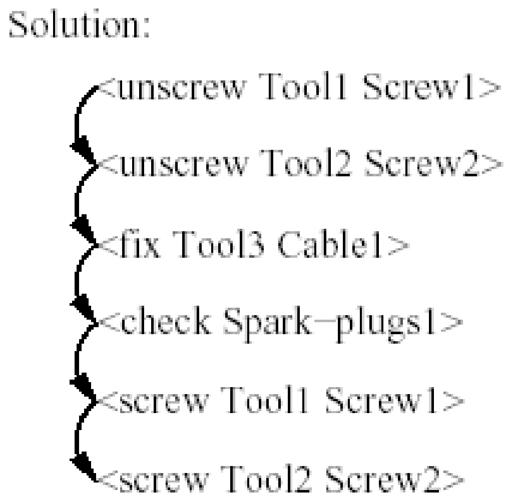



**Figure 10:** A Possible *Car Fixing* Plan.

A set of problems that use the same operators is called a `domain`. The goal of MAPWeb is to give solutions to problems in a domain as just defined.

The sequence MAPWeb follows to solve a problem is like this:

1. The user interacts with a UserAgent to define his/her problem. Then the UserAgent sends to a PlannerAgent an achieve performative containing the problem definition.

2. The PlannerAgent receives the problem definition and analyzes it. Usually, a user problem contains a lot of detail and that makes problem solving computationally very expensive for classical AI planning systems. For that reason, before attempting to solve it, the PlannerAgent discards some of the detail and transforms the user problem into an abstract representation. For instance, in the car fixing domain, there could be many different kinds of parts and tools to deal with them. In that case, the PlannerAgent would reduce the number of different parts and tools to a manageable quantity. Then, the user problem would be transformed into an abstract representation that uses only the reduced set of parts and tools. At this point, the PlannerAgent would use a planning system to solve the abstract problem and get several possible abstract solutions. However, the user needs all the details to be able to apply the plan. Furthermore, many of the abstract solutions might not be valid in reality since they ignore actual details. Therefore, the abstract plans have to be completed and validated. The PlannerAgent analyzes which parts of the abstract plans require completion, and asks for details the WebAgents.

3. WebAgent receives PlannerAgent's queries for details, looks for information at those web-sites the agent is specialized in, and returns the information to the PlannerAgent in a common shared format. If it cannot find the requested information, the PlannerAgent will be informed, and it will discard all the plans that include the invalid operator. For instance, different car companies could maintain web-sites with information on technical characteristics of cars, tools, and parts, which could be used by the specialized WebAgents to fill in the requested details. If the WebAgents could not find information for validating the `fixing` step, because, for instance, there are no `Tools` to handle `Cable1`, all the plans that include this step will be discarded by the PlannerAgent.

4. Finally, the PlannerAgent receives a tell performative from several WebAgents, validates and completes the abstract plans, and returns complete plans to the UserAgent. In our example, a possible complete solution would include which actions to perform and the specific tools and parts to use. This plan could be utilized directly by the user.

## 5. Illustrative Application of MAPWeb

In principle, MAPWeb can be applied to many and diverse problem solving domains. In this section, we describe how MAPWeb has been applied to a particular domain –**"electronic tourism"** (or simply *e-tourism*) – and how the different agents cooperate to solve problems in this domain. Earlier versions of MAPWeb have been described by Camacho and coauthors.[49] This section will first present the e-tourism domain (i.e., how solutions are represented) and then how the different agents in MAPWeb cooperate to provide solutions to the user. Communication between the UserAgent, the PlannerAgent, and the WebAgents will be elaborated in detail.

### *5.1. Electronic Tourism Domain*

An e-tourism system has to provide the following services to the user:

1. Informing how to go from the initial to the destination town using different means of transportation.

2. Lodging at destination.

3. Informing about possibilities when visiting a town (renting a car, local transport, etc.).

4. Informing how to return to the initial (or other) town.

MAPWeb has the abilities enumerated above. However, in this paper, we will focus mainly on the logistics problem of providing the user with plans to move from one place to another. Moving from place to place involves long-range trips that can be accomplished via airplanes, trains, or buses. It also involves taking local transportation means (taxi, subway, bus, etc.) to move between airports, bus stations, or train stations. In order to represent and provide solutions to the user, we have defined an e-tourism domain that uses the operators illustrated in Table 2.

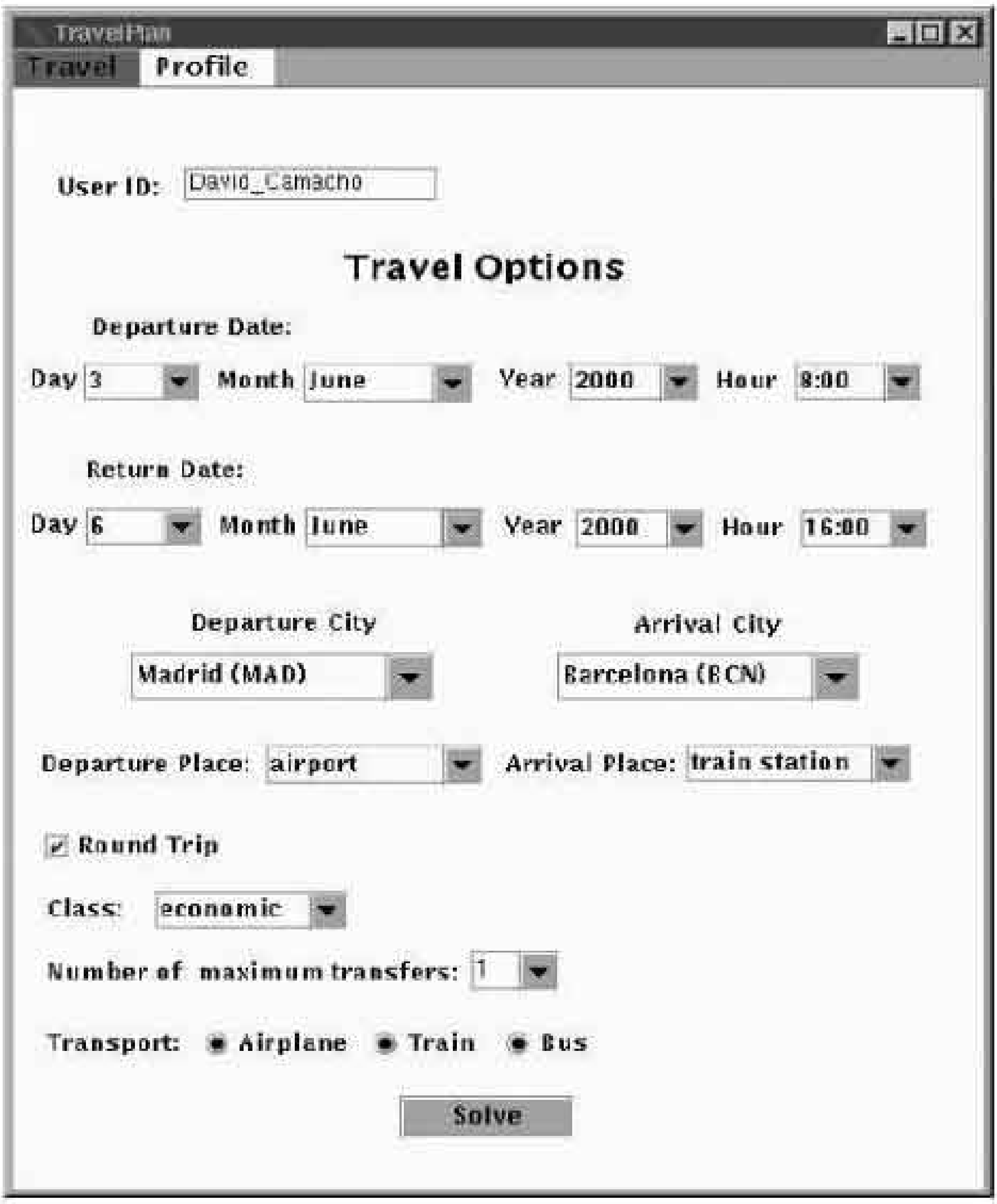
**Table 2:** E-tourism planning operators

| **Operator** | **Arguments** |
|---|---|
| TRAVEL-BY-AIRPLANE | User-name, Company, Origin-airport, Destination-airport |
| TRAVEL-BY-TRAIN | User-name, Company, Origin, Destination |
| TRAVEL-BY-BUS | User-name, Company, Origin, Destination |
| MOVE-BY-LOCALBUS | Origin, Destination |

| | |
|---|---|
| MOVE-BY-TAXI | Origin, Destination |
| MOVE-TO | Origin, Destination |
| BOOK-HOTEL-ROOM | User-name, Hotel, City |
| … | |

### *5.2. UserAgent → PlannerAgent Communication*

The UserAgent provides a gui to the user, so that s/he can describe the problem and the restrictions associated with it. Obviously, guis depend heavily on the problem domain: other domains would require other guis. Figure 11 presents the *input*-gui to the system.



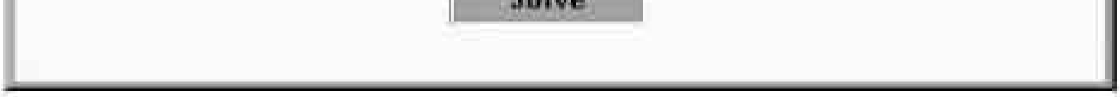


**Figure 11:** User Agent Input.

The data the user has to provide to the system is as follows:

- Departure and return dates
- Departure and arrival cities
- Starting and arrival places inside the cities (airport, train station, bus station, etc.)
- One-way or return trip
- Maximum number of transfers
- Cost (economy class, business class, first class, second class, tourist class, etc.)
- Long-range transport (airplane, train, or bus)

In the example given in Figure 11, the user plans to travel to Barcelona (Spain) from Madrid (Spain) on the 3rd of June at 8 o'clock. The return date is the 6th of June at 4 or later. The user would like to start his/her trip from an airport and wishes to end it at a train station in Barcelona. S/he wants to minimize cost and s/he does not specify the long-range transportation means. Also, s/he does not want to transfer more than once.

Once the UserAgent has received the problem, it sends an achieve performative to a PlannerAgent and waits for the solution.

### *5.3. PlannerAgent → WebAgents Cooperation*

The PlannerAgent receives from the UserAgent a problem and proceeds with building an abstract representation that retains only the parts essential for the planning process. For instance, a typical description of the previous problem for an AI planning system would include:

- All the cities in the world
- All the airports, train stations, etc. inside those cities
- All the plane, bus, and train companies in the world
- All local transportation means (taxi, subway, etc.) in the cities

Any classical AI planning system would get bogged down if it tries to find a plan by considering all these elements. Instead, the PlannerAgent builds an abstract problem in the following way:

1. First, it defines an abstract city. This city includes all the possible local transport and only the long-range transport terminals that the user wishes to use. For instance, if the user wants to travel only by plane, the abstract

city would include just airports. The goal is to reduce the number of elements in the problem, so that the planner can handle them more efficiently. In the previous example, as there are no restrictions on the long-range transport, the abstract city would have airports, bus stations, and train stations.

2. Then, this abstract city is repeated as many times as is the maximum number of transfers supplied by the user. It is important to note that the cities are abstract cities (i.e. they have no attached names; they are present in the abstract plan to represent the initial, middle, and final travel points).

3. Finally, the rest of the details provided by the user are ignored at this stage. For example, departure and arrival times, travel cost, etc. is not considered. This data will be used later to query the WebAgents and validate the abstract solutions.

As an illustration, from the problem given by the UserAgent, the PlannerAgent would construct a planning problem that includes three unnamed cities: `city0`, `city1`, and `city2`. `city0` is the departure city, `city2` is the destination, and `city1` is a (possible) transfer city. Each of the cities includes all possible local transportation means, abstract locations (`hotel1`, …) and terminals (`airport0`, `trainstation0`, ...). Finally, the planning problem would include an initial situation of the user being at `airport0` in `city0`, and the goal situation is that of the user being at `trainstation2` in `city2`.

The above described abstract problem would be given to the PlannerAgent planner (Prodigy4.0) which would obtain several possible abstract solutions. In this case, the planner would reply with the plans given in Figure 12.

Solution 1:

<move-to trainstation0 bustop01>

<move-to bustop01 airport0>

<travel-by-airplane user1 plane0 airport0 airport1>

<move-to airport1 bustop11>

<move-by-localbus bustop11 bustop12>

<move-to bustop12 trainstation1>

<travel-by-train user1 train1 trainstat1 trainstat2>

Solution 2:

<move-to trainstation0 bustop01>

<move-by-localbus bustop01 bustop02>

<move-to bustop02 airport0>

<travel-by-airplane user1 plane0 airport0 airport2>

<move-to airport2 bustop21>

<move-by-localbus bustop21 bustop22>

<move-to bustop22 trainstat2>

**Figure 12:** Two abstract solutions generated by Prodigy for the travel problem.

This is a set of abstract plans that contain no details. Some of the steps in the plan might not even exist in the real world. Therefore, these plans need to be validated and completed. This is achieved by querying the WebAgents. In this case, the following query schemas would be generated:

Queries:

(travel-by-airplane user plane0? Madrid city1?)

(travel-by-train user train1? city1? Barcelona)

The queries above have some uninstantiated variables (`plane0?`, `train1?`, and `city1?`). The variable `city1?` will be instantiated by the PlannerAgent before querying the WebAgents. The PlannerAgent will choose several actual cities by using some heuristics. For every selected city, an actual query will be generated. For instance, the first query schema would be translated into:

Queries:

(travel-by-airplane user plane0? Madrid Valencia)

(travel-by-airplane user plane0? Madrid Alicante)

These queries (and all the additional information given by the UserAgent) are sent to several WebAgents that are specialized in airplane travel, so that variable `plane0?` is instantiated as well.

A WebAgent receives a query and associated with it data and transforms it into an actual web query. The WebAgent is familiar with the structure of the data stored at the web sites it is specialized in, and it knows how to look for information in these web sources. The retrieved information is then analyzed and stored in a common template, which is subsequently sent to the PlannerAgent. In our example, the information in Table 3 would be returned to instantiate the variable `plane0?`. Actually, that variable can be instantiated in many different ways, as many as the possible flights from Madrid to Valencia.

**Table 3:** Retrieved WebAgent Information.

| **Information-Flights** | **flight1** | **flight2** | **flight3** |
|---|---|---|---|
| air-company | Iberia | Iberia | Spanair |
| http-address | www.iberia.es | www.iberia.es | www.spanair.com |
| flight-id | 323 | 450 | null |
| ticket-fare | 38200 | 21850 | 43700 |
| currency | ESP | ESP | ESP |
| flight-duration | Null | null | null |
| airport-departure-city | MAD | MAD | MAD |
| departure-date | 03-06-00 | 03-06-00 | 03-06-00 |
| airport-arrival-city | VLC | VLC | VLC |
| return-date | 06-06-00 | 06-06-00 | 06-06-00 |
| class | D | D | null |
| number-of-passengers | 1 | 1 | 1 |
| round-trip | one-way | one-way | one-way |

Finally, the PlannerAgent instantiates all the abstract plans for which it has received from the WebAgents a positive answer for each plan step. Those plans in which one or several steps received either no answer or an empty answer are rejected. Therefore, only plausible plans are sent back to the UserAgent. Every abstract plan will be instantiated into many different actual plans. Table 4 shows two of the generated solutions.

**Table 4:** Solutions given by MAPWeb.

| **Solution1** | **Solution2** |
|---|---|
| (move-to trainstation0 bustop01)<br>(move-to bustop01 MAD)<br>(travel-by-airplane SMejias Iberia MAD VLC)<br>(move-to VLC bustop11)<br>(move-by-localbus bustop11 bustop12)<br>(move-to bustop11 VLCtrainstation1)<br>(travel-by-train Smejias Talgo VLC BCN)<br>(move-to BCN bustop21)<br>(move-by-localbus bustop21 bustop22)<br>(move-to bustop22 hotel2) | (move-to trainstation0 bustop01)<br>(move-by-localbus bustop01 bustop02)<br>(move-to bustop02 MAD)<br>(travel-by-airplane SMejias plane0 MAD BCN)<br>(move-to BCN bustop21)<br>(move-by-localbus bustop21 bustop22)<br>(move-to bustop22 hotel2) |

### *5.4. PlannerAgent → UserAgent Communication*

Finally, the UserAgent receives the list of actual plans and presents them to the user. Figure 13 shows the *output*-gui where the found plans for our problem are displayed. If the user wants more information about a plan step, s/he can click on the corresponding operator and get data about departure time, location, etc.

## 6. Conclusions

We have presented a multi-agent approach (MAPWeb) to solve planning problems using the information available on the Web. In particular, this paper focuses on how to solve user planning problems by means of cooperation between a PlannerAgent and several WebAgents. This cooperation amounts to dividing the planning problem into two parts: generation of abstract plans (by the PlannerAgent) and validation-completion of these plans (by the WebAgents). This is done since planning problems contain a lot of details that makes the classical AI problem solving computationally very expensive.

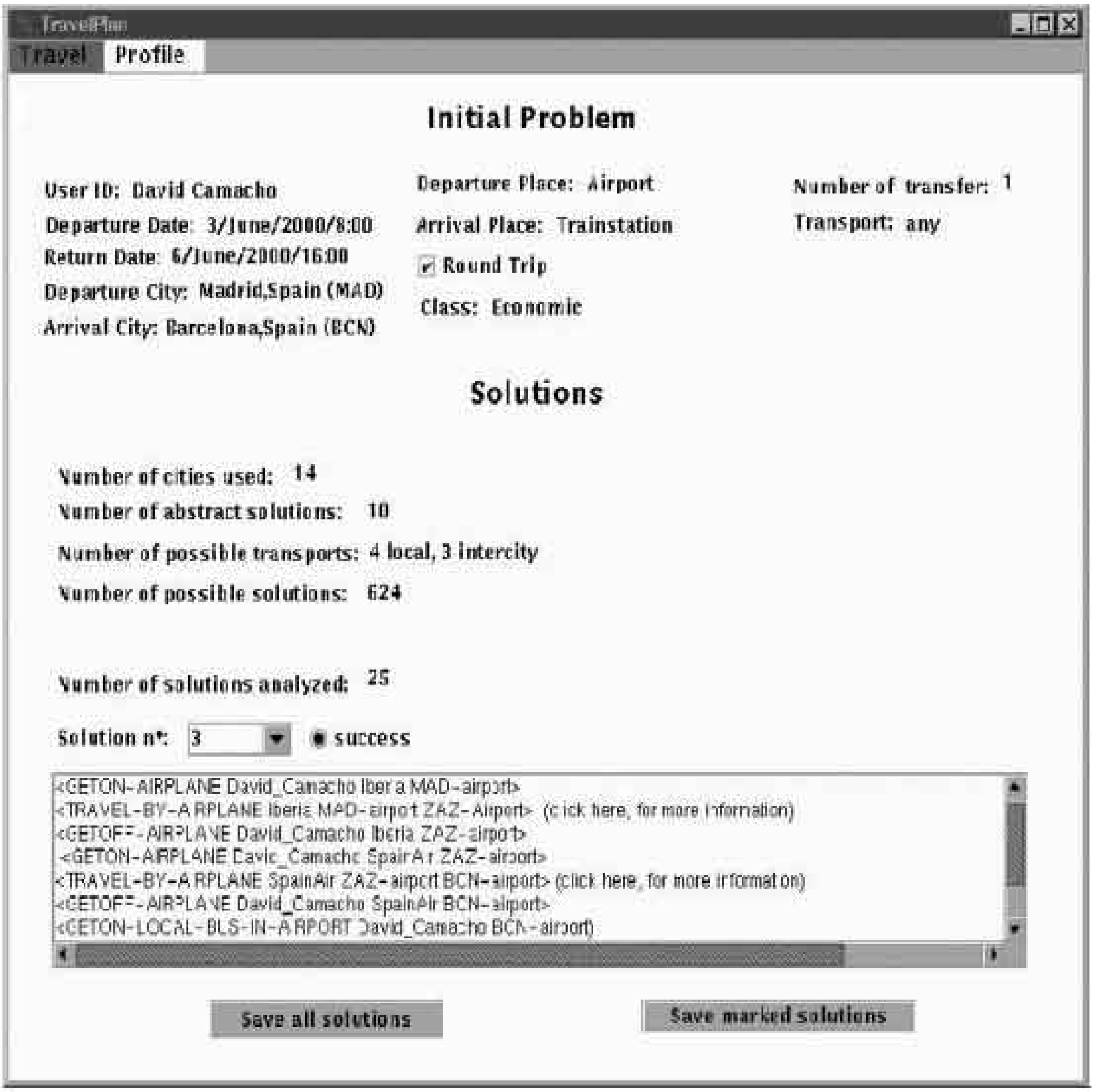



**Figure 13:** UserAgent Output.

Therefore, before attempting to solve a planning problem, the PlannerAgent discards some of the details and builds an abstract, easier to solve, version. Then, several abstract solutions are obtained. However, many of the abstract solutions might not be valid in reality due to the fact that some of the actual details are ignored. Therefore, the abstract plans have to be completed and validated.

There is another important reason to divide the planning process. Information on the Web is heterogeneous and is provided in multiple formats. Therefore, it makes sense to have many different agents specialized in each information source or web site. Thus, WebAgents not only free PlannerAgents from the details, they also isolate them from the complexity of the information sources.

MAPWeb is not only a set of conceptual ideas. The described architecture has been implemented. Also, it has been applied to an actual domain (e-tourism) where the cooperation characteristics described above are fully exploited.

## 7. Future Directions

Some of the lines of future work include:

- Cooperation between several PlannerAgents. In many planning domains, a problem can be divided into a set of sub-problems. Each sub-problem could be sent to a different PlannerAgent. This would be useful for two reasons. First, problem solving can be parallelized. And, second, different kinds of sub-problems could be sent to specialized PlannerAgents that might use different planning techniques.

- Reuse of information stored in both PlannerAgents and WebAgents. Agents can learn from experience. For instance, if a PlanningAgent has previously solved a problem, it can be stored in an internal database for later use, either by the same agent or by others. In a similar manner, a WebAgent can reuse information retrieved previously to reduce the Web access.

- Application of Case-Based Reasoning techniques,[50] so that new planning problems can be solved by adapting the plans from previously solved similar problems. This would reduce enormously the planning process which is computationally very expensive.

- Finally, in order not to overload the user with too many plans, MAPWeb should be able to rank the solutions and recommend the best ones using user profiles and by learning from user's previous behavior.

---

**Notes:**

1. Jose Luis Ambite and Craig A. Knoblock, "Agents for information gathering," in *IEEE Expert: Intelligent Systems and their Applications* (September/October 1997); Craig A. Knoblock and Jos• Luis Ambite, "Agents for Information Gathering," in *Software Agents*, ed. J. Bradshaw (Menlo Park, CA: AAAI/MIT Press, 1997).

2. Oren Etzioni, "Moving up the information food chain," *AI Magazine* 18, 2 (1997): 11-18.

3. Michael N. Hunhs and Munindar P. Singh, *Readings in Agents* (San Francisco: Morgan Kaufmann, 1997).

4. J. Bradshaw, ed., *Software Agents* (Menlo Park California: AAAI Press, 1997); Michael R. Genessereth and Steven P. Ketchpel, "Software agents," *Communications of the ACM* 37, 7 (1994): 48-53.

5. W. Brenner, R. Zarnekow, and H. Wittig, *Intelligent Software Agents. Foundations and Applications*, (New York, Springer-Verlag, 1998); Michael Wooldridge and Nicholas R. Jennings, "Intelligence agents: Theory and practice," *Knowledge Engineering Review* (October 1994).

6. Ricardo Aler, Daniel Borrajo, and Pedro Isasi, "Genetic programming and deductive-inductive learning: A multistrategy approach," in *Proceedings of the Fifteenth International Conference on Machine Learning, ICML'98*, ed. Jude Shavlik (Madison, Wisconsin, July 1998), pp. 10-18.

7. Manuela M. Veloso and Jim Blythe, "Linkability: Examining causal link commitments in partial-order planning," in *Proceedings of the Second International Conference on AI Planning Systems* (Chicago, IL: AAAI Press, June 1994), pp. 170-175.

8. D.E. Wilkins and D.L. Myers, "Multiagent planning architecture," in *Proceedings on The Fourth International Conference on Artificial Intelligence Planning Systems AIPS98* (June 1998).

9. Katya P. Sycara, “Multiagent systems,” *AI Magazine* 18, 2 (1998).

10. Alan H. Bond and Less Gasser, *Readings in Distributed Artificial Intelligence* (San Francisco California: Morgan Kaufmann, 1988); Les Gasser, “An Overview of Distributed Artificial Intelligence,” *Distributed Artificial Intelligence: Theory and Praxis* (Kluwer Academic Publishers, 1992), pp. 9-30.

11. Nicholas R. Jennings, *Coordination Techniques for Distributed Artificial Intelligence* (O'Hare et al., 1996), pp. 187-210.

12. Brenner, Zarnekow, and Wittig, *Intelligent Software Agents.*

13. Bradshaw, *Software Agents*; Michael N. Hunhs and Munindar P. Singh, *Readings in Agents.*

14. Hyacinth S.Nwana, “Software agents: An overview,” *Knowledge Engineering Review* 11, 3 (October/November, 1996): 205-224.

15. Wooldridge and Jennings, “Intelligence agents: Theory and practice.”

16. R. Armstrong, D. Freitag, T. Joachims, and T. Mitchell, “Webwatcher: A learning apprentice for the world wide web,” in *Working Notes of the AAAI Spring Symposium: Information Gathering from Heterogeneous, Distributed Environments* (Stanford University: AAAI Press, 1995), pp. 6-12.

17. H. Lieberman, “Letizia: An agent that assists web browsing,” in *Proocedings of the International Joint Conference on Artificial Intelligence IJCAI95* (1995), pp. 924-929.

18. Armstrong, Freitag, Joachims, and Mitchell, “Webwatcher.”

19. Erik Selberg and Orentz Etzioni, “The Metacrawler architecture for resource aggregation on the web,” *IEEE Expert* (IEEE, January/February 1997): 8-14.

20. Brenner, Zarnekow, and Wittig, *Intelligent Software Agents*; Wooldridge and Jennings, “Intelligence agents: Theory and practice.”

21. Chelliah Thirunavukkarasu, Tim Finin, and James Mayfield, “Secret agents - a security architecture for the KQML agent communication language,” in *Proceedings of the ACM CIKM Intelligent Information Agents Workshop* (New York: Association of Computing Machinery, December 1995).

22. Michael R. Genessereth and Steven P. Ketchpel, “Software agents;” Wooldridge and Jennings, “Intelligence agents: Theory and practice.”

23. Orentz Etzioni, N. Lesh, and R. Segal, “Building softbots for UNIX, in *Software Agents-Papers from 1994 Spring Symposium,* Technical Report SS-94-03 (AAAI Press, 1994), pp. 9-16.

24. Michael Coen, *SodaBot: A Software Agent Environment and Construction System*, Technical Report 1493 (MIT AI Lab, 1994).

25. Yigal Arens, Craig A. Knoblock and Chun-Nan Hsu, “Cooperating agents for information retrieval,” in *Proceedings of the Second International Conference on Cooperative Information Systems* (Toronto, Ontario, Canada: University of Toronto Press, 1994); Craig A. Knoblock and Jos• Luis Ambite, *Software Agents*.

26. J.L. Ambite and C. A. Knoblock, “Planning by rewriting: Efficiently generating high-quality plans,” in *Proceedings of the Fourteenth National Conference on Artificial Intelligence* (1997); Craig A. Knoblock, Steven Minton, Jose Luis Ambite, and Naveen Ashish, “Modeling web sources for information integration,” in *Proceedings of the Fifteenth National Conference on Artificial Intelligence* (Madison, WI, 1998).

27. Oren Etzioni, “Moving up the information food chain.”

28. Erik Selberg and Orentz Etzioni, “The Metacrawler architecture for resource aggregation on the web.”

29. H. Lieberman, “Letizia: An agent that assists web browsing.”

30. Thorsten Joachims, Dayne Freitag, and Tom Mitchell, “A tour guide for the world wide web,”in *Proceedings of IJCAI97* (August 1997); R. Armstrong, D. Freitag, T. Joachims, and T. Mitchell, “Webwatcher: A learning apprentice for the world wide web.”

31. J. Hullen, Ralph Bergmann, and F. Weberskirch, “Webplan: Dynamic planning for domain-specific search in the internet,” in *Workshop Planen und Konfigurieren PuK-99* (1999).

32. Chelliah Thirunavukkarasu, Tim Finin, and James Mayfield, “Secret agents - a security architecture for the KQML agent communication language.”

33. Reticular Systems, *AgentBuilder. An Integrated Toolkit for Constructing Intelligent Software Agents* (Reticular Systems Inc., February 1999).

34. Deepika Chauhan and Albert D. Baker, “Jafmas: A multiagent application development system,”in *Proceedings on The Second International Conference on Autonomous Agents Agent's 98* (Minneapolis, May 9-13, 1998).

35. <http://www.fipa.org> (27 May 2001).

36. Jeremy Pitt and Fabio Bellifemine, “A protocol-based semantics for FIPA’97 acl and its implementation in Jade,” in *Proceedings of AI*IA* (1999); Fabio Bellifemine, Agostino Poggi, and Giovanni Rimassa, “Jade - a fipa-compliant agent framework,” in *Proceedings of PAAM’99* (London, April 1999), pp. 97-108.

37. Charles Petrie, “Agent-based engineering, the web, and intelligence,” *IEEE Expert* 11, 6 (December 1996): 24-29.

38. Anthony Chavez and Pattie Maes, “Kasbah: An agent marketplace for buying and selling goods,” in *Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology* (London, UK, April 1996).

39. David Wilkins and Karen Myers, “A common knowledge representation for plan generation and reactive execution,” *Journal of Logic and Computation* 5, 6 (1995): 731-761; D.E. Wilkins and D.L. Myers, “Multiagent planning architecture.”

40. Manuela Veloso, Tucker Balch, and Scott Lenser, “Integrating information agents, planning, and execution monitoring,” in *Proceedings of Agents-2000* (June 2000).

41. J. Baxter and R. Hepplewhite, “A hierarchical distributed planning framework for simulated battlefield entities,” in *Proceedings of 19th Workshop of the UK Planning and Scheduling Special Interest Group PLANSIG 2000* (December 2000).

42. Any MAPWeb-agent is built using a set of standard and reusable Java packages and classes implemented as the basis to build the different system agents.

43. Tim Finin, Jay Weber, et. al., *Draft specification of the KQML agent communication language* (1993); Tim Finin, R. Fritzson, D. Mackay, and R. McEntire, “KQML as an agent communication language,”in *Proceedings of the Third International Conference on Information and Knowledge Management CIKM94* (New York: Association of Computing Machinery, 1994), pp. 456-463.

44. This characteristic is being developed at the moment.

45. This module is being developed at the moment.

46. Manuela M. Veloso and Jaime G. Carbonell, “Derivational analogy in PRODIGY: Automating case acquisition, storage, and utilization,” *Machine Learning* 10, 3 (March 1993): 249-278; J. Hullen, Ralph Bergmann, and F. Weberskirch, “Webplan: Dynamic planning for domain-specific search in the Internet.”

47. M. Veloso, J. Carbonell, A. Perez, D. Borrajo, E. Fink, and J. Blythe, “Integrating planning and learning: The Prodigy architecture,“ *Journal of Experimental and Theoretical AI* 7 (1995): 81-120.

48. Finin and Weber, *Draft specification of the KQML agent communication language.*

49. David Camacho, Daniel Borrajo, and Jose Manuel Molina, “Travelplan: A multiagent system to solve web electronic travel problems,” in *Workshop on Agent-Based Recommender Systems. Fourth International Conference on Autonomous Agents* (Barcelona, Catalonia Spain: ACM, June 2000); David Camacho, Jos• M. Molina, and Daniel Borrajo, “A multiagent approach for electronic travel planning,” in *Proceedings of the Second International Bi-Conference Workshop on Agent-Oriented Information Systems AOIS-2000* (Austin, TX, USA: AAAI-2000).

50. Agnar Aamodt and Enric Plaza, “Case based reasoning: Foundational issues, methodological variations and system approaches,” *AI Communications* 7, 1(1994): 39-59; Manuela M. Veloso and Jaime G. Carbonell, “Derivational analogy in PRODIGY: Automating case acquisition, storage, and utilization.”

---

**DAVID CAMACHO** is a lecturer at the Department of Computer Science, Universidad Carlos III de Madrid (UC3M). He is a member of the Complex and Adaptive Systems Group (SCALab). He holds a B.Sc. in Physics Science (1994) from the Universidad Computense de Madrid. He has researched in several areas, including planning, inductive logic programming, fuzzy logic, and multi-agent systems. He has also participated in international projects on automatic machine translation and optimization of industrial processes. *Address for correspondence:* Universidad Carlos III de Madrid, Avda. de la Universidad, 30, 28911 Legané s (Madrid), Spain, E-mail: dcamacho@ia.uc3m.es

**JOSÉ M. MOLINA** received his Ph.D. in Telecommunication Engineering from the Universidad Politecnica de Madrid, in 1997. He has been a member of the System, Signal and Radio Communications group in the University Politecnica of Madrid since1992. He is an Associate Professor at the Computer Science Department, Carlos III of Madrid University. He also participates in the Complex Adaptive Systems group. He has published more than 5 journal and 50 conference papers. His current research focuses on the application of soft computing techniques (Neural Networks, Evolutionary Computation, Fuzzy Logic and Multi-agent Systems) to engineering problems, such as RADAR, robot control and vision. *Address for correspondence:* Universidad Carlos III de Madrid, Avda. de la Universidad, 30, 28911 Legané s (Madrid), Spain, E-mail: jmolina@ia.uc3m.es

**DANIEL BORRAJO** has been University Professor in Computer Science at the Universidad Carlos III de Madrid (UC3M) since 1998. Previously, he was Associate Professor at UC3M and Universidad Polit•cnica de Madrid (UPM). He received his Ph.D. in Computer Science from UPM in 1990. He holds a B.Sc. in Computer Science (1987) from the Universidad Polit•cnica de Madrid. He has been vice-dean of the Computer Science degree at UC3M and, currently, he is the head of the Department of Computer Science. He has published over 50 journal and conference papers. Dr. Borrajo’s main research interest is the integration of different Artificial Intelligence techniques, especially concerning machine learning. His main research interest now is focusing the learning system towards real-world planning domains and problems, considering the quality of the solutions, and the self-experimentation of the system. *Address for correspondence:* Universidad Carlos III de Madrid, Avda. de la Universidad, 30, 28911 Legané s (Madrid), Spain, E-mail: dborrajo@ia.uc3m.es

**RICARDO ALER** is a lecturer at the Department of Computer Science, the Universidad Carlos III. He has researched in several areas, including automatic control knowledge learning, genetic programming, and machine learning. He has also participated in international projects on automatic machine translation and optimization of industrial processes. He holds a PhD in Computer Science from Universidad Polit•cnica de Madrid (Spain) and a M.Sc. in Decision Support Systems for Industry from Sunderland University (UK). He graduated in Computer Science at Universidad Polit•cnica de Madrid. *Address for correspondence:* Universidad Carlos III de Madrid, Avda. de la Universidad, 30, 28911 Legané s (Madrid), Spain, E-mail: aler@inf.uc3m.es

**BACK TO TOP**

---

**Information & Security. An International Journal**
**e-mail: infosec@mbox.digsys.bg**

# MAPWEB: Cooperation between Planning Agents and Web Agents

*David Camacho, Jose M.Molina, Daniel Borrajo, Ricardo Aler*

**Keywords:** Information System, Agent Architecture, Multi-Agent Systems, Web Agents, Intelligent Agents, Planning

**Abstract:** The paper presents MAPWEB (Multi-Agent Planning on the Web), a multi-agent system for cooperative work between different intelligent software agents whose main goal is to solve user planning problems using the information stored in the World Wide Web (WEB). MAPWEB is made of heterogeneous mixture of intelligent agents whose main characteristics are cooperation, reasoning, and knowledge sharing. The architecture of MAPWEB uses four types of agents: UserAgents that are the bridge between the users and the system; ControlAgents (Manager and Coach Agents) that are responsible for managing the rest of the agents; PlannerAgents that are able to solve planning problems; and, finally, WebAgents whose aim is to retrieve, represent, and share information obtained from the WEB. MAPWEB solves planning problems by means of cooperation between PlannerAgents and WebAgents. Instead of using the PlannerAgent to solve the whole planning problem, the PlannerAgent focuses on a less restricted (and therefore easier to solve) problem (what we call an abstract problem) and cooperates with the WebAgents to validate and complete abstract solutions. In order for cooperation to take place, a common language and data structures have also been defined.

**full text**

Authors: **Stanimir Stojanov, Roumen Venkov and Radi Radev**
Title: **An Agent-Based Approach to the Development of Information Systems for Military Logistics**
Year of issuance: **2002**
Issue: **Information & Security. Volume 8, Number 2, 2002, pages 239-251**
Hard copy: **ISSN 1311-1493**

---

# AN AGENT-BASED APPROACH TO THE DEVELOPMENT OF INFORMATION SYSTEMS FOR MILITARY LOGISTICS

Stanimir STOJANOV, Roumen VENKOV and Radi RADEV

---

**Table Of Contents:**



---

## 1. Introduction

The development of modern information systems in various application areas (e-commerce, logistics, and intelligent telecommunication services) presents a serious challenge to software technologies. This can be acknowledged to the difficulty, complexity, heterogeneity and distributed functionality of the systems. It is still a difficult task to find a homogeneous technological framework for the effective solution of these problems. Different models and standard frameworks have been developed for this kind of systems.[1] Although they posses certain advantages, the requirement for generality makes their application in specific domains relatively expensive. In addition, considerable resources are needed for their adaptation.[2] Agent-based systems solve some of these problems, especially those systems developed in e-commerce applications.[3] The agent-based technology offers flexibility and easier adaptation.

To provide efficient development facilities, modern software technologies have to meet different, often contradictory, requirements. The point here is that they have to strike a balance between user requirements and required technological support. From this point of view, the technology is expected to effectively solve the following two problems:

- To support the execution of the distributed functionality of the systems, and
- To provide the user with easy and intuitive interface for work with the system, while the internal complexity and heterogeneity of the system remains hidden.

The approach presented in this paper is characterized by the following two phases:

1. Definition of a model that meets the specific requirements and reflects the organization of the military logistics in the Bulgarian Army;
2. Use of the agent-based framework MALINA for the implementation of a prototype based on the model.

Following these two stages, we intend to improve the system performance.

## 2. A Model of the Military Logistics Support

The following general considerations form the basis for development of a model for military logistics. From a user's perspective, the distributed information system could be considered as an architecture consisting of two parts: a server and a client. The server provides a given functionality accessed trough requests to the system. The processes performed are transparent to the users. The client part is the interface between users and the system. In order to facilitate the interaction, it is necessary to provide a powerful technological support, comprised of a multi-layered distributed architecture, application software (implemented as objects, components or autonomous agents), standardized internal interfaces, and relevant control mechanisms.

### *2.1. Services*

In the adopted approach the concept *service* is a basic notion. From one point of view, the service is a specific interface which users use for interaction with the system. Furthermore, using this concept, we intend to strike a balance between user and technological aspects of a complex distributed system.

From a user's perspective the service is a logically separated functionality and understanding and implementing it does not create problems. However, it is not an easy task to define the services from a technological point of view. Adapting the service architecture to conform to the requirements of our approach, involves difficulties such as flexibility, transparency and adaptability of the program components. Moreover, the components have to operate in a dynamic environment and they are under the control of different types of management mechanisms (centralized or decentralized). In this context, we define the services as generic structures, consisting of two components:

- Kernel, that realizes the functionality (business-logic) of the service;

- Shell, containing all necessary software modules that allow the execution of the service in an operational environment.

The services can be implemented as objects, components or agents. The specification of standardized, logically separated functionality depends on the particular application area. For instance, in e-commerce the standardized functions are specified by Electronic Commerce Building Blocks (ECBB), [4] while in telecommunication Service Independent building Blocks (SIBs) are used.[5,6]

### *2.2. Model overview*

Generally, the systems for military logistics are rather complex. They are built on heterogeneous platforms, and often it is necessary to distribute their functionality. To deal with the complexity associated with military logistics, we have decomposed the problem. The elaborated relevant and stable architecture and the distributed functionality are specified and presented on the logical model below (refer to Figure 1).

The model consists of three independent *horizontal levels*. These levels specify the static functionality of the system and reflect various degrees of generalization. The information systems of the individual units function on the *low level*. The *middle level* is formed by the information systems of the formations and the corps. The *high level* includes the systems of the Armed Forces and the General Staff.

Each level is built by a separate logistical chain. A chain integrates the activities, which are logically interrelated and applied to homogeneous objects. The following four types of logistical chains are defined in the model:

- *Material Chains* (MC) – information activities that organize, provide and control the movement of materials;
- *Financial Chains* (FC) – activities for management and control of the financial flows;
- *Personnel Resources* (PR) – information activity related to human resources management;
- *The Document Chains (DC)* – activities for organization and management of the documents.

The document processing chains are considered at a more general level. They are specific logistical chains with two functions. First, they are used for horizontal integration of the other three chains. Second, they have their own contents and specifics and they can be integrated vertically between the levels.

Within a given level, we allow integration of the logistical chains (we shall name it as *horizontal* integration). The horizontal integration can be automated and realized by an adequately integrated information system. Horizontal integration can be achieved between different logistical chains.

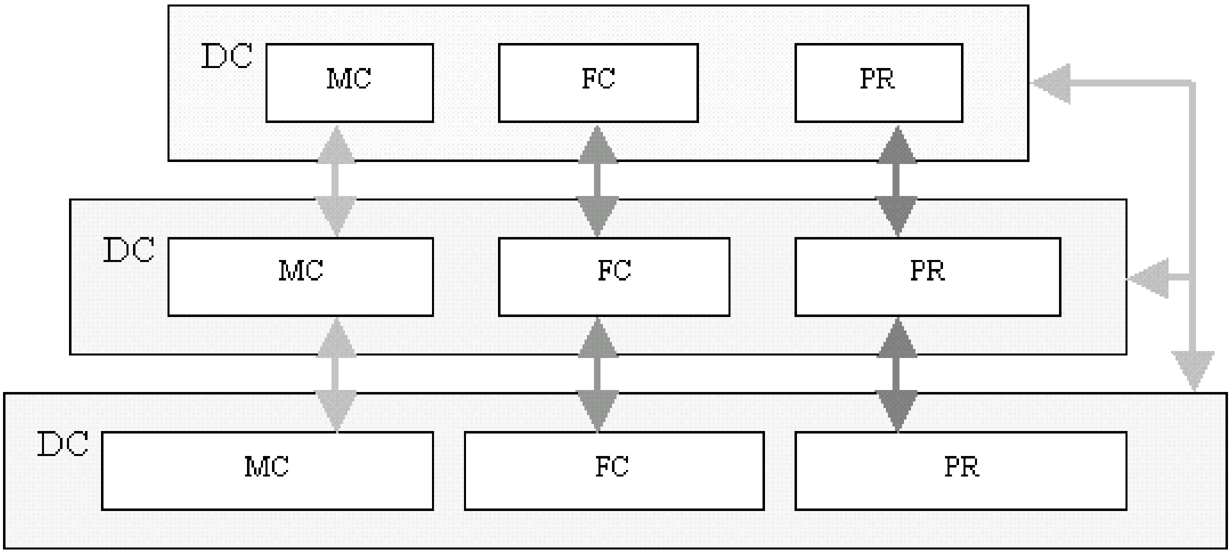



**Figure 1**

Between the different levels we also allow for the so-called *vertical* integration. The vertical integration is possible only between homogeneous logistical chains (chains of one type). This type of integration can be automated and serviced by the information systems of a corresponding application area. The *application area* is defined as a collection of the related homogeneous logistical chains of the different levels. The logistical chains of the different levels are integrated by means of standardized *interfaces.*

In the model, the operation and control of the programming modules is organized as *business-transactions*. It is possible to define separate standard business-transactions, which could be parameterized, customized, and modified. All this provides a satisfactory level of flexibility in the system. The business-transactions can be *horizontal* (within the framework of a technological chain or an integrated system) or *vertical* (within the framework of one or more applications). Horizontal business-transactions are, for example, the instance management and control of the *Food Supply Service*, depot facilities, the orders and deliveries, the financial services in the units. Vertical business-transactions are the management and control of the orders and deliveries in the Bulgarian Army, the management of the depot (warehouse) facilities at formation and corps level.

## 3. MALINA Technology

For development of the system we have applied a multi-agent technology known under the name MALINA (Multi-Agent Local Integrated Network Associations). A thorough description of the technology can be found in various publications.[7,8,9,10] The technology supports a bottom-up approach to the development of multi-agent applications.[11] The MALINA technology specifies three levels in the development process:

- *Abstract level* – a hypothetical infrastructure, which provides the theoretical framework of the technology, is defined at this level.

- *Conceptual level* – concepts are decomposition and specification of the hypothetical infrastructure; they aid the development of supporting programming tools and an appropriate development environment. The following four concepts form the basis for establishing the development environment of the technology:

- *Abstract Agent Architecture* – it specifies the architecture of an abstract generic agent, by which the agents in different applications can be created;

- *AgentCities* – it specifies a static infrastructure for multi-agent applications;

- *AgentAssociations* – it provides various possibilities for building agents associations;

- *MobileServices* – it provides different modes for automatic generation of mobile agents;

- *Development level* – it is composed of the supporting tools and the development environment of the adopted technology.

### *3.1. Abstract Agent*

This concept has two goals: to specify a basic architecture, which can be used as a generic framework for the creation of application agents and to define the agents' life cycle. The concept defines an architecture, which will be known as *Abstract Agent*, by means of which the agents in a particular application can be created. The Abstract Agent (see Figure 2) is a generic abstract structure $A = \langle \pi,\phi,\mu \rangle$, where:

- $\pi$ - denotes the *agent engine*, which includes the agent's interfaces to the environment (sensors and effectors) and the agent's local control;

- $\phi$ - is the *specialization* of the agent. This is an abstract interface to some software modules that implement the agent's functionality

- $\mu$ - represents the *mentality* of the agent. This in turn specifies also an abstract interface to a well structured knowledge base. The knowledge base consists of independent segments. Each segment includes knowledge and processing methods that define the agent's behavior, for example beliefs, intentions, and commitments.

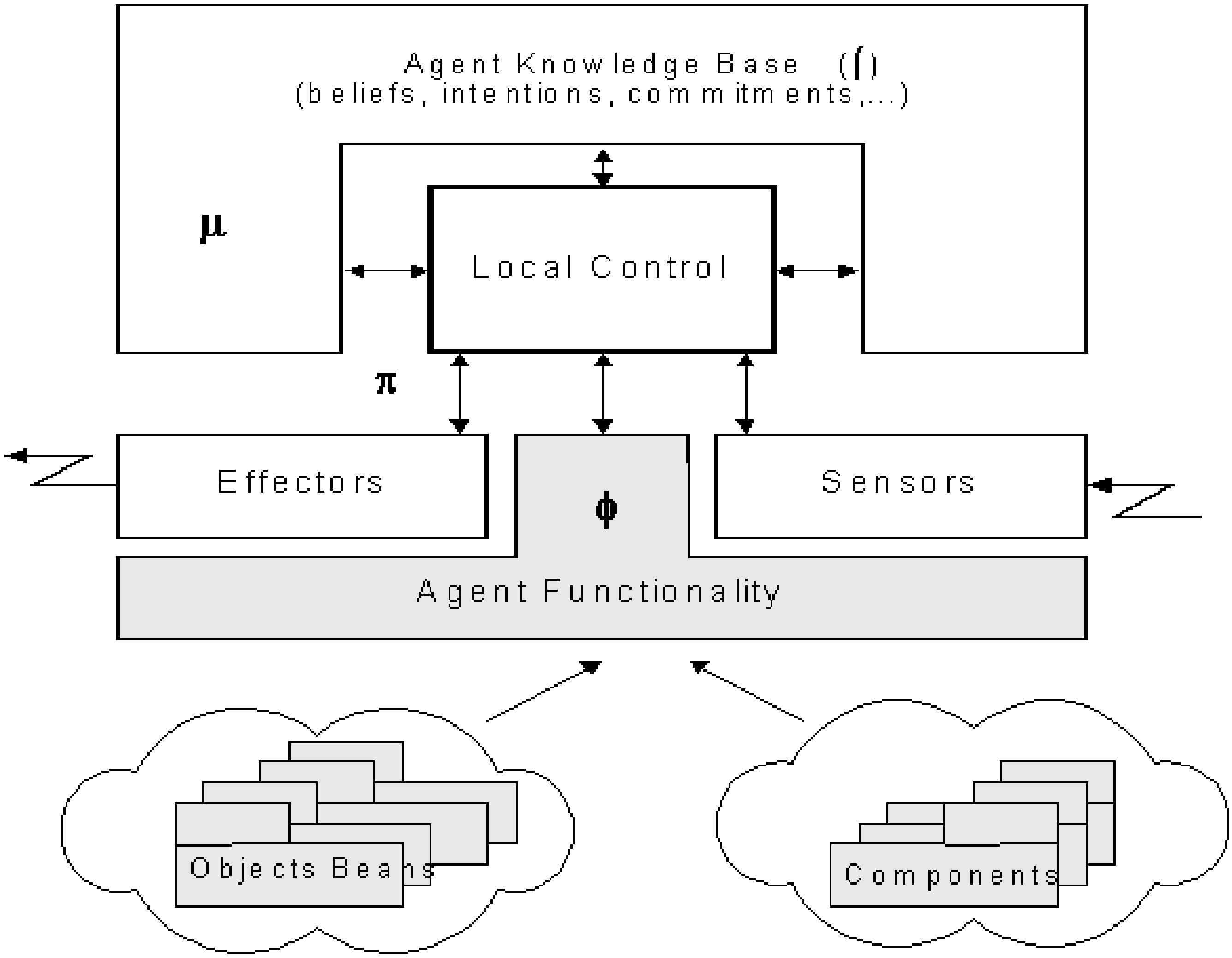



**Figure 2**

The Agent's Life Cycle is an abstract procedure that includes the following steps:

- Observe an input at time T
- Record any relevant information
- Check input consistency
- Solve the problem or generate a plan
- Compose answer expression
- Send the answer

***3.2. Agent Cities***

The concept *Agent Cities* specifies an open infrastructure, by which the multi-agent system can be modeled. The basic building block of the infrastructure is the "*city*". The "cities" are local virtual areas where the agents of the system will be located. The infrastructure is static, i.e. the "cities" have constant locations (addresses). The agents' locations within the "cities" are also static. In order to identify and control the "cities" and the agents, the concept defines and supports two address spaces – a logical address space and a physical address space.

Within the logical space the “cities” are identified by names (identifiers), which are unique in the system. The logical address of agents is described by a pair *(Name, Social role)*. The agent names (identifiers) are unique within the “cities”. The logical address space is visible to users and system designers/developers. The agents in an application interact with their logical addresses.

The physical address space is used to support and control the messaging processes on a computer network. This address space is visible only during the run-time of a multi-agent application. To users and designers it is transparent.

### *3.3. Agent Associations*

In conformity with the technology a multi-agent system is the set of all specified “cities”. On the other hand, the set of the located agents in a “city” does not constitute an agent association. The interactions between the agents in a “city” are a source of additional semantics that cannot be defined only by the agents themselves. An agent association is specified further by the type of agents’ interaction and by the organizational and management models of the “cities”. In this concept, we distinguish three types of interactions between agents:

- *Communication* – the simplest possible interaction form, which supports the message exchange between the agents;

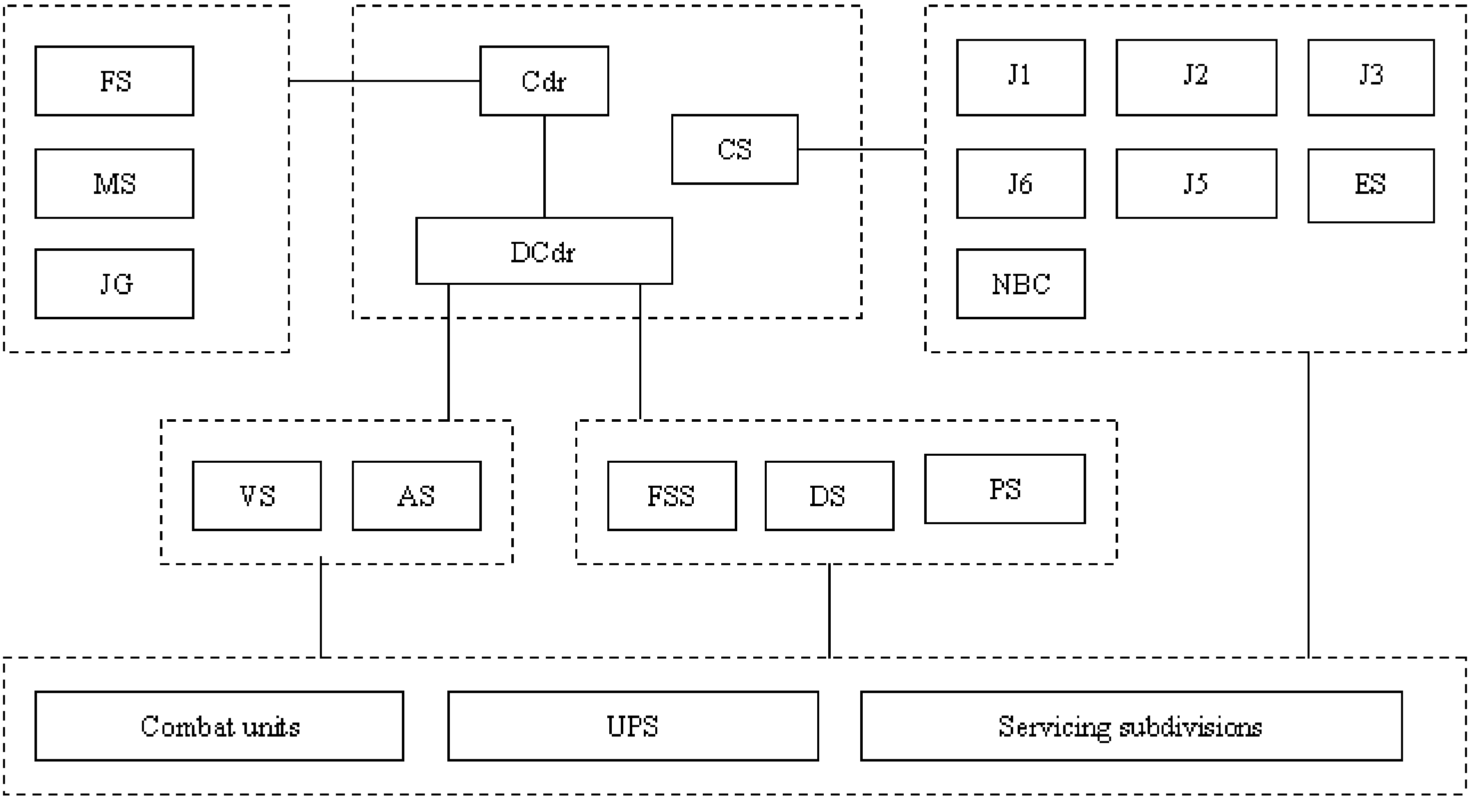
- *Synchronization* – a more complex interaction form, which can be used for synchronization of the agents. Synchronization is necessary for solving joint tasks. This type of interaction is applicable only between agents that show intelligent behavior (the so-called intelligent agents);

- *Cooperation* – the most complex form of interaction. It is used for preparation and planning of common work. It is applicable only for intelligent agents.

In order to implement the interaction concept we use the agent communication language KQML.[12] The technology supports two major types of control mechanisms in the “cities”: centralized and decentralized management models. In the case of *centralized management* only a single intelligent agent is present in the “city” and it plays the role of a manager. The rest of the agents (the so-called workers) do not exhibit intelligence and they are subordinate to the manager. The manager receives the requests, distributes the work, and determines the responsibilities of the workers (i.e. it can be viewed as a kind of planner agent). In the case of d*ecentralized management* there are more intelligent agents present in the “city”. In order to accomplish the requirements of a received request, the agents have to cooperate.

## 4. DIANA – A Multi-Agent Application in Military Logistics

DIANA is a prototype of the information system for logistics support that implements the low level of the model described above. During system development, the model presented in Figure2, has been adapted to the organizational scheme of the real logistical system of the units. The system is designed as a three-layered architecture. The *first* layer is the client interface to the system, which enables the users (i.e., commanders, deputies of the commanders, the chiefs of the directorates and services, the warehousemen (the chiefs of the depots), the clerks, the suppliers (mess-sergeants), and others) to interact with the system. It controls the correctness of the incorporated information, generates the request and sends it to the application part of the system. The *second* layer is divided into two sub-layers – system and application. The system sub-layer controls and checks the processing of the different business transactions. It interprets the received order, generates the corresponding business transaction and controls its execution. This sub-layer supports also the horizontal integration of the system. The application layer contains separate program components, which realize the business logic of the different logistical chains (Figure 3). The database contains necessary basic nomenclatures.




LEGEND:

Cdr( Commander)

FS – Finance Section

DS – Dress Service

CS ( Chief of Staff)

DCdr (Deputy Commander for Logistics)

MS ( Medical Service)

PS ( Petroleum, oil and lubricants Service)

JG ( Jurisconsult General)

VS (Vehicle Service)

AS (Weapon Service)

ES (Engineer Section)

FSS (Food Supply Section)

NBC (NBC Service)

UCP(Units for Combat Provision)

Directorates:

J1( Personnel)

J2(Intelligence)

J3(Operative)

J5(Plans and Policy)

J6 (Communications-Electronics)

**Figure 3**

The module "*Management and Control of the Food Supply Service*" has been implemented in the first version of DIANA. The module is designed as a multi-agent application with three groups of agents: system agents (SA), application agents (AA) and personal agents (PA). The SAs and AAs are server-based agents, i.e. they are transparent to the users. The business logic of the module is distributed among the functional kernels (the specialization) of the application agents. Some of the application agents are listed below:

- $AA^{MG}$ – the agent is a generator of the menu-ration. In addition, it justifies and optimizes the created menu-ration;

- $AA^{DM}$ – this agent controls the document processing in the system
- $AA^{RG}$ – the agent is responsible for preparation of different reports
- $AA^{WM}$ – a warehouse (food depot) manager
- $AA^{DB}$ - this is a set of filter agents, which provide the access to the database.

The system agents assist in the operation of the application agents, where their primary task is to support the communication between the agents, the addressing of the agents and the management of the agent associations (cities). The users do not have a direct access to the system and to its information resources. The relations between the user and the server part of the system are realized by personal agents. They are visualized and they play different social roles (according to the terminology of MALINA), such as *Commander*, *Chief of the Food Supply Service*, *Clerks*, *Chief of the Food Depot*. With the help of the development tools (Figure 4) of MALINA we can configure also other PAs or customize the SAs and AAs in accordance with the requirements of any new application.

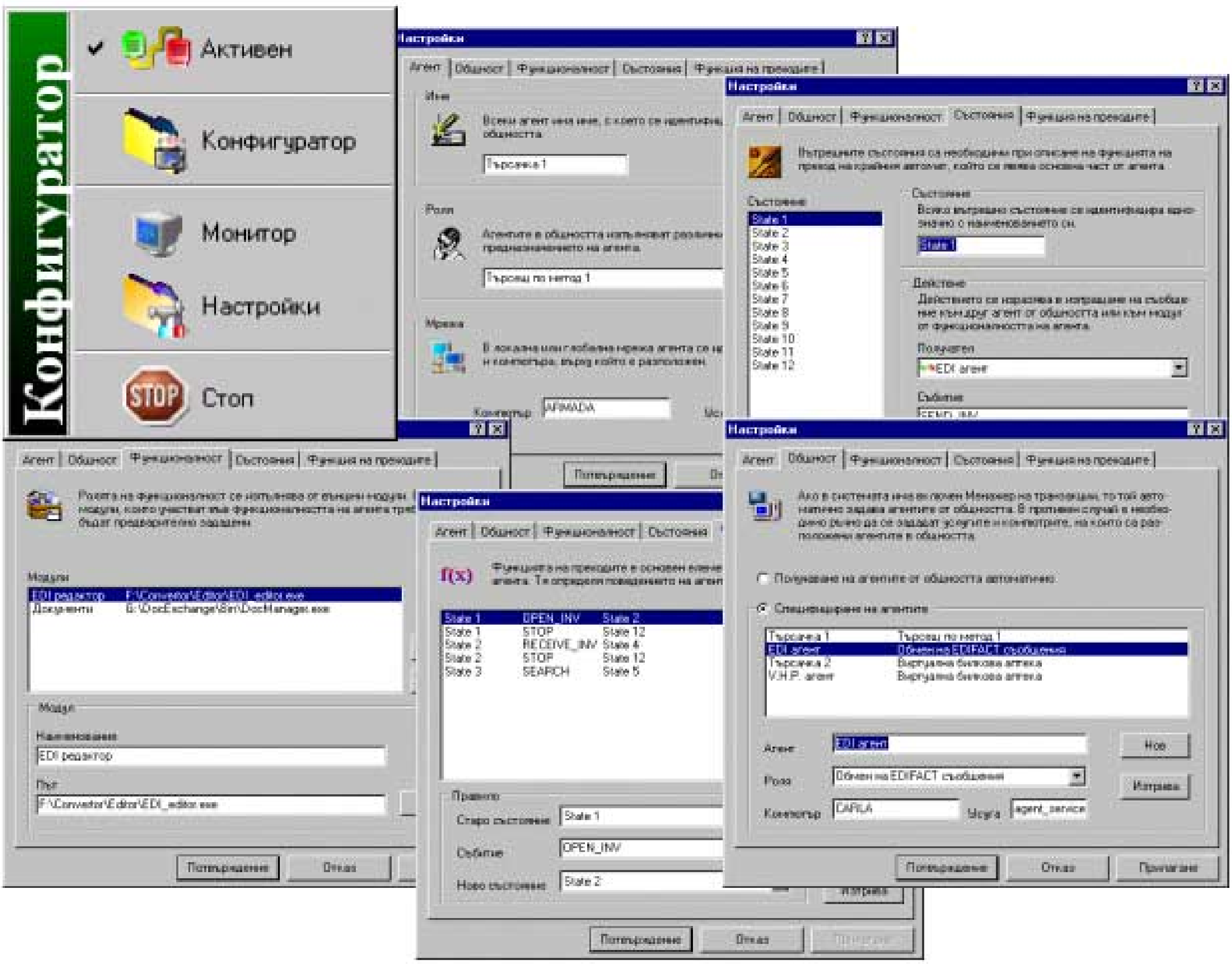



**Figure 4**

The DIANA-agents are deployed in the four “cities” (in the terminology of MALINA) described below, which form the infrastructure of the module. It is possible to make different arrangements. Figure 5 shows an infrastructure, which have been used during the testing of the system:

- $C_{FSM}$ - Food Service Management
- $C_{Cdr}$ – Commander ( that only controls and signs documents)
- $C_{WM}$ - Warehouse Management
- $C_{DBS}$ – Database Services and Document Processing Services

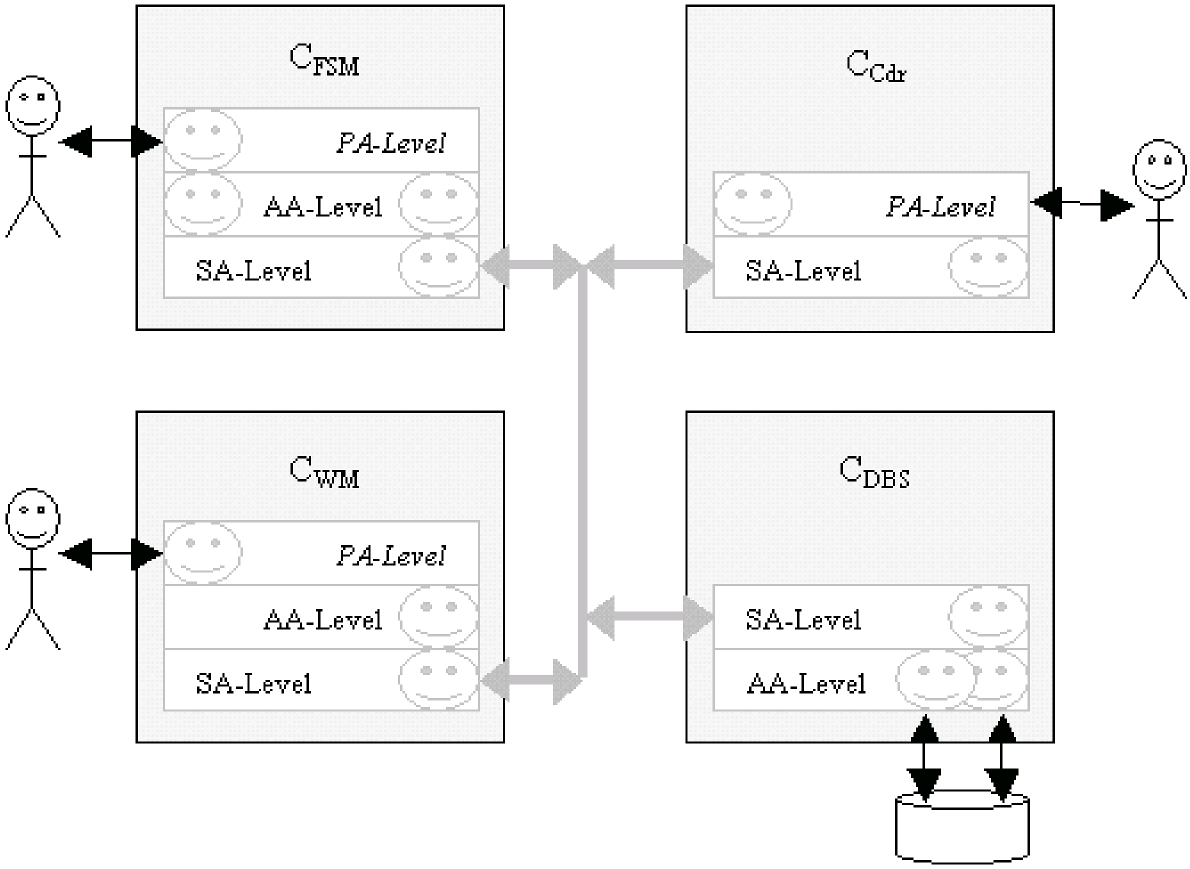



**Figure 5**

Figure 6 illustrates the administration module, by which we can create and support the information sources in the system.

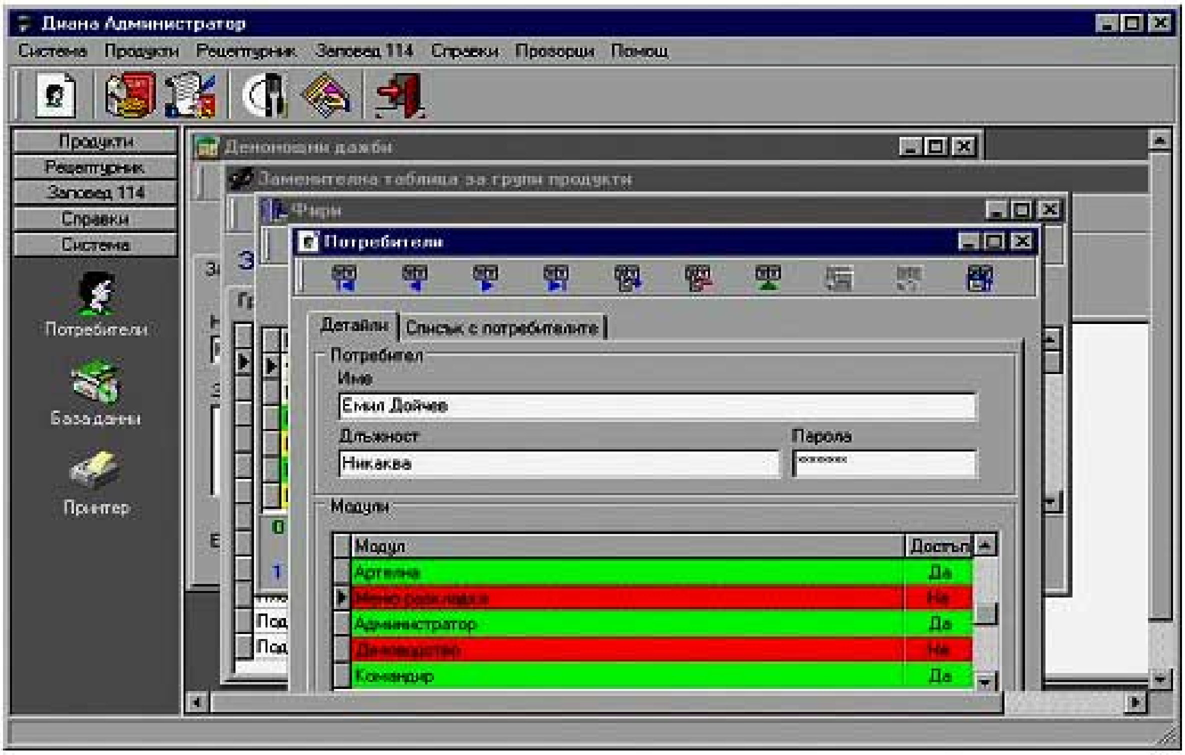



**Figure 6**

## 5. Conclusions

DIANA – the proposed system for military logistics – is implemented by means of the development tools of the MALINA technology on a J2EE platform and SQL Server. Experiments are performed using a system prototype in the unit 34560 in Yambol and in the Navy Base of Bourgas. The results of these initial experiments with the prototype demonstrate that the agent-based methodology is a viable approach to the development of military information systems.

---

**Notes:**


1. IBM, "San Francisco Product-Information," <http://www.ibm.com/Java/Sanfrancisco>, (1998); OMG, "CORBA BOCA – Business Object Component Architecture," *OMG-Document Nr. Bom/98-01-07*, (1998).

2. M. Merz, *Electronic Commerce*, (Dpunkt.verlag, 1999).

3. *BargainFinder*, <http://bf.cstar.ac.com/bf>; *Firefly*, <http://www.firefly.com>; *Jango*, <http://www.jango.com>; *Kasbah*, <http://kasbah.media.mit.edu>; *PersonaLogic*, <http://www.personalogic.com>; *Tete-a-tete*, http://ecommerce.media.mit.edu/tete-a-Tete (June 2002).

4. EBES/EWOS Project Team on Building Blocks for Electronic Commerce (PRB), "Building Blocks for Electronic Commerce," *Final Report*, (Brussels, 1997).

5. Q. Kong, G. Chen, and N. Lawler, "Distributed Architecture for IN Service Management," *IEEE Intelligent Network Workshop IN'96*,( The Carlton Crest Hotel, Melbourne, Australia, April 21-24, 1996).

6. SIEMENS, "IN Introduction, Reconfigured Services," (Vienna: IN Training Center, 1997); SIEMENS, "IN Introduction, System Overview and Functions," (Vienna: IN Training Center, 1997).

7. C. Wille, R. Dumke, and S. Stojanov, "Performance Engineering in Agent-Based Systems – Concepts, Modeling and Examples," in *Current Trends in Software Measurement, Proceedings of the 11th International Workshop on Software Measurement*, (Montreal,

Canada, August 28-29, 2001), pp. 153-181.

8. S. Stojanov, M. Kumurdjieva, “MASTT - Technology and Its Application in Electronic Commerce Systems,” in *Proceedings of the Workshop- Concurrency, Specification & Programming*, (Berlin: Humboldt-University, September 28-30, 1998).

9. S. Stojanov, M. Kumurdjieva, E. Dimitrov, and A. Schmietendorf, “Technological Framework for Development of Agent-based Applications,” in *Proceedings of the Workshop - Concurrency, Specification & Programming*, (Berlin: Humboldt-University, October 9-11, 2000), pp. 299-311.

10. S. Stojanov, I. Ganchev, “A multi-agent technology for creating of information systems for electronic commerce,” in *Proceedings of the Second South Eastern European Conference on e-commerce*, (Sofia, 24-26 October 2000), pp. 92-96.

11. H.D. Burkhard, “Theoretische Grundlagen in der verteilten Kuenstlichen Intelligenz,” in *Verteilte kuenstliche Intelligenz*, ed. J. Mueller, (Germany: BI-Mannheim, 1993), pp. 157-189.

12. T. Finin, J.Weber, G.Wiederhold, M.Genesereth, “Specification of the KQML Agent-Communication Language” *The DARPA Knowledge Sharing Initiative External Interfaces Working Group*, (University of Toronto, 1994); Y. Labrou, T.Finin, “A Proposal for a new KQML Specification”, (University of Maryland Baltimore County, 1997).

---

**STANIMIR STOJANOV** is the head of the E-Commerce Laboratory at the University of Plovdiv. His scientific interests are in the fields of multi-agent technologies and systems, software architectures, intelligent telecommunication services, and E-Commerce. He is author of more than 50 papers and textbooks. *Address for correspondence:* Faculty of Mathematics and Informatics, University of Plovdiv, Plovdiv, Bulgaria; Phone: (++359 32) 277 292; E-mail: stani@uni-plovdiv.bg

**ROUMEN VENKOV** is the general manager of ISY Intellect Ltd. His interests are in the fields of project management, configuration management, VL databases, and artificial intelligence. He is author of more than 30 papers and studies. *Address for correspondence:* 12 N.Gogol Str••t, Sofia, Bulgaria; Phone: (++359 2) 943 36 44; E-mail: isyint@bul.evro.net

Colonel **RADI RADEV** is the ex-commander of the unit 34560 in Yambol where the model of the Military Logistician Support DIANA was realized. *Address for correspondence:* Druzhba-2, bl. 317, entr. 2, ap. 1, Sofia, Bulgaria, Phone: (++359 2) 983 83108.

**BACK TO TOP**

---


**Information & Security. An International Journal**
**e-mail: infosec@mbox.digsys.bg**

# An Agent-Based Approach to the Development of Information Systems for Military Logistics

*Stanimir Stojanov, Roumen Venkov and Radi Radev*

**Keywords:** Software technology, Military Logistics, Models, Multi-Agents Applications

**Abstract:** The paper presents an approach to the development of multi-agent applications based on the MALINA-technology. The aim is to create a general technological framework for development of modern information systems for military logistics and E-commerce. First, a general model for military logistics is developed. Then, the paper presents the system architecture and a prototype of the agent-based system DIANA.

**full text**

Author: **Information & Security**
Title: **I&S Research Centers**
Year of issuance: **2002**
Issue: **Information & Security. Volume 8, Number 2, 2002, pages 255-256**
Hard copy: **ISSN 1311-1493**

---

# E-COMMERCE LABORATORY (ECL)

## Faculty of Mathematics and Informatics
## Plovdiv University “Paisii Hilendarski”

The E-Commerce Laboratory was created at the end of 1998. The Laboratory is part of the Faculty of Mathematics and Informatics at the Plovdiv University “Paisii Hilendarski,” Plovdiv, Bulgaria. The main purpose of the laboratory is to integrate more closely research and higher education.

The laboratory is active in three main areas:

- Support to the education of graduate students in the masters’ program “Advanced Information Technologies.”
- Research oriented towards creation of advanced object-, component- and agent-oriented development tools.
- Experimentation with the developed tools in pilot projects and design of prototypes in the fields of e-Learning, e-Commerce and Intelligent Telecommunications Services.

One of the high priority activities of the laboratory is the creation and support to scientific contacts with European universities, laboratories and research groups. The ECL Laboratory actively seeks opportunities for participation in international projects jointly with foreign partners. The Laboratory has established close contacts with the Humboldt University, the Berlin University and the Technical University – Magdeburg (Germany).

Currently, the research and development efforts of the Laboratory are concentrated on three pilot projects:

- *Culture, historical heritage and natural landmarks of Bulgaria.* The goal of the project is to develop an intelligent portal and supporting services for presentation of the cultural and historical heritage of Bulgaria. Various e-Catalogues are developed within the framework of this project.

- *XCTL* – a reengineering project. The main goal of this project is to provide students with hands-on experience in this new field of Software Engineering. This project is being developed jointly by seven universities from Germany, Bulgaria, Yugoslavia and Macedonia.

- *Virtual University* – an effort to support the education of students by providing various e-services over the Internet. The project includes development of new component- and agent-oriented architectures.

Additional information is available at the ECL Website: http://ecl.pu.acad.bg/

---

**BACK TO TOP**

---


**Information & Security. An International Journal**
**e-mail: infosec@mbox.digsys.bg**