RAIRO-Oper. Res. 44 (2010) 307–322
DOI: 10.1051/ro/2010019

RAIRO Operations Research
www.rairo-ro.org

SELF-ADAPTIVE AIR-SEA SIMULATION BASED ON MULTI-SENSORS AGENTIFICATION

S. Peyruqueou[1], D. Capera[1], T. Médina[2] and C. De Murcia[2]

Abstract. Combat Management System training uses simulation of an overall tactical situation. This involves the real-time management of numerous and diverse entities to keep the simulation scenario consistent in a highly dynamic environment. To address this difficult problem, we propose an adaptive multi-agent system in which each entity is considered as a smart sensor/effector mobile. The autonomy and the dynamic behaviour offered to each entity leads the simulation to self-adapt to inevitable disturbances of the user. According to the cooperation paradigm, this approach also allows the mobiles to highlights a coherent global behaviour with mutual helping. Finally, the system shows the relevance of the Emergence Technologies in the elaboration of a new generation of sensors. This software is currently under development in GATES, a project of the DCNS company.

Keywords. Emergence technologies, multi-agent systems, smart sensors, self-adaptation, cooperation, real-time simulation.

Received March 3, 2010. Accepted November 8, 2010.

[1] UPETEC, 10 avenue de l'Europe – 31520 Ramonville Saint Agne, France.
[sylvain.peyruqueou;davy.capera]@upetec.fr
[2] DCNS, SIS/DDP, BP 403 – 83055 Toulon Cedex, France.
[thierry.medina;cyril.de-murcia]@dcnsgroup.com

Résumé. L'entraînement sur les Systèmes de Gestion de Combat requiert la simulation de théâtres d'opération complets. Cela implique la gestion en temps réel d'entités nombreuses et de types variés afin de conserver le scénario de la simulation cohérent malgré un environnement très dynamique. Pour aborder ce problème difficile, nous proposons un système multi-agent adaptatif dans lequel chaque entité mobile est considérée comme un capteur/effecteur intelligent. L'autonomie et le comportement dynamique apportés à chaque entité amène la simulation à s'adapter automatiquement aux inévitables perturbations induites par l'utilisateur. Suivant le paradigme de la coopération, cette approche permet également aux mobiles de mettre en évidence un comportement global cohérent avec aide mutuelle. Au final, le système tend à montrer la pertinence des Technologies de l'Emergence dans l'élaboration d'une nouvelle génération de capteurs. Ce logiciel est actuellement en développement dans le cadre de GATES, un projet de la société DONS.

Mots Clés. Technologies émergentes, système, multi-agent, capteurs intelligents, auto-adaptation, coopération, simulation en temps réel.

Mathematics Subject Classification. 68U20, 91A12.

1. Introduction

Since the SENIT 6 combat management system (CMS), DCNS provides on board training (OBT) capability as a means of achieving an optimal operational use of the combat systems capabilities. Making use of a computer generated Forces (CGF), the OBT aims at providing an overall tactical situation in order to assess and improve the CMS operator skills (trainees). Due to several factors such as the multiplicity of actors (*e.g.* in collaborative training exercise with several connected CMS platforms), the real time execution of the scenario differs from what it was first planned or foreseen. Consequently, the trainer is often overloaded as a result of many corrections performed on line to keep control of the scenario.

The global need raised in this paper is to provide automatisms at the simulation level in order to reduce the trainer tasks. The simulated actors (called mobiles) involved in the scenario have to keep a consistent behaviour in the whole dynamic environment. Each one has different objectives and evolves with a limited perception of its environment, thus it is obviously impossible to globally elaborate a reaction and apply it to all of them. Besides, in the case of large-scale battles, mobiles are numerous and various, these are very diverse (all kinds of ships but also aircrafts and submarines) and some of them can appear dynamically like planes from aircraft-carriers. Moreover, maritime areas imply long response times for ships so it is important to anticipate as best as possible.

No classical approach offers the required potentialities for such a challenge and that is why DCNS has called upon Upetec to design together a system that implements Emergence Technologies as part of the GATES project.

Instead of addressing the problem globally, the adaptive multi-agent approach proposed by Upetec considers mobiles of the simulation as autonomous entities. Each one becomes a smart sensor/effector involved in a complex system. Using the paradigm of the cooperation, they make emerge a coherent collective behaviour based on their individual perceptions, decisions and actions.

In addition, this design takes into account the need to manage heterogeneous data and to share them between all smart sensors/effectors. This enables to cross the partial nature of the information a mobile-sensor collects and, on the other hand, to best adapt its behaviour towards others.

To demonstrate the relevance of this approach, this paper first highlights two examples developed in the GATES project. A presentation of the system design follows in a second part. We then shortly explain the adaptive multi-agent systems principles (AMAS) and their implementation to design the simulator. Finally, we present related works and prospects to extend this technology in the air-sea simulation area and in complex systems with highly integrated sensors.

2. GATES realistic simulations

The integration of a first multi-agent software module in GATES yet allows to simulate realistic collective behaviours. The simple examples presented below are intentionally reduced to few mobiles for a clearer explanation. It should be however noted that the system remains perfectly coherent in tests with more entities.

2.1. Fleet with a consistent behaviour

Figure 1 presents the following scenario:

- the cargo A aims to follow a complex trajectory;
- frigates B and C are equipped with a detection system and have to escort the cargo from the rear.

Throughout the course of the cargo, the following frigates continuously adapt themselves to avoid a collision with each other and with obstacles.

In this example, each mobile-sensor gets multiple local information. Thanks to their data processing and decision capacities, the frigates are able to combine their own position, those of other mobiles and those of obstacles in order to move in the best way to achieve their goal.

2.2. Cooperative mobiles

Figure 2 shows the following scenario:

- the cargo A is equipped with a detection system but not a weapon system and its purpose is to follow its route;

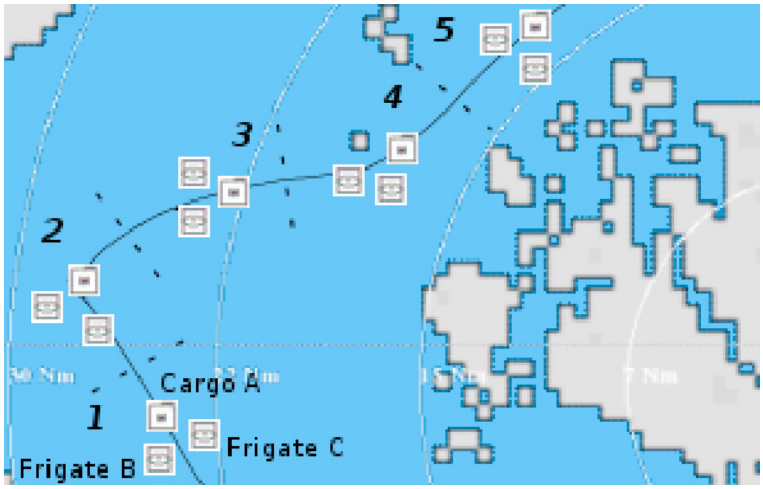

FIGURE 1. Frigates escorting a cargo.

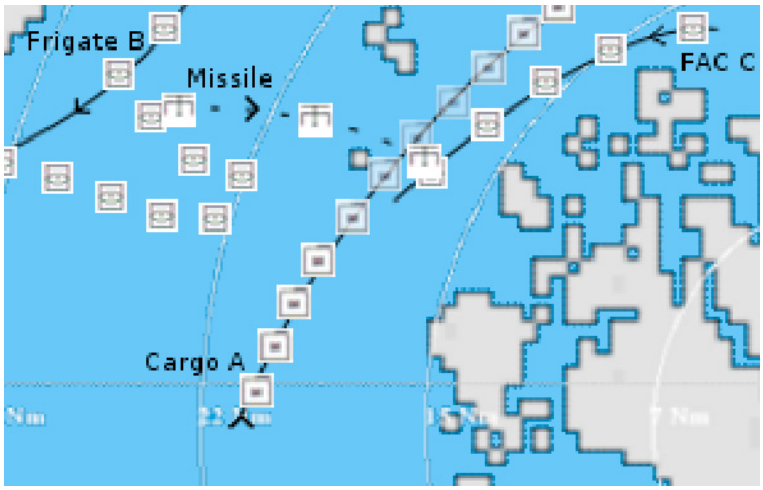

FIGURE 2. Interception on request.

- the frigate B, ally of the cargo A, has a weapons system and aims at continuing its route;
- the FAC C is a pirate ship that plans to attack cargo A.

The cargo A detects the FAC C when this last enters its radius of perception (Step 2). It identifies the FAC as a threat but is not able to neutralize it in that A has no weapons system. The cargo then asks its allies to help it.

The frigate B responds positively to the request, leaves its trajectory and neutralizes the FAC C using a missile (Steps 3–5) before resuming its route (Steps 6 to 10).

In this simple case, the sensor and the effector are disjoint. Through this example, we want to illustrate the cooperation capabilities provided by entities of an AMAS. The cargo holds information but cannot do something by itself, thus it sends a request to be helped. Thanks to this collaboration mechanism, the cargo's request can be treated by the frigate even if, at first, it did not perceive the threat by itself.

To do that, the system design involves the addition of communication capabilities to each mobile. This includes:

- a communication system to send and receive messages;
- the ability to standardize its perceptions so that all its allies are able to understand each other.

2.3. System analysis

In the first example, each entity is a smart sensor/effector involved in a dynamic environment out off any supervision. It simply and clearly illustrates the relevance of Emergence Technologies for a collective resolution.

This case of escort illustrates that the coherence of the global behaviour dynamically appears from the behaviour of individuals both adaptive and autonomous. They operate optimally out off global supervision. This also permits to dynamically add or remove sensors/effectors without any other change.

The cooperation shown by the second example leads to an additional capability of the global system. If a sensor fails (here, the frigate which can not perceive the FAC), it can be compensated by other sensors (the cargo). This makes the global system less vulnerable to failures. In industrial applications it ensures the system to continue to operate in degraded mode until a sensor is repaired.

With the AMAS approach, we finally get all the required elements for a new generation of sensors. First, they are autonomous and adaptive. Then they process information locally and can normalize it to communicate with others. Finally, they are able to optimize the overall behaviour of the system to meet the need in the most coherent way, even if some components are experiencing failure.

3. Self-organizing agents mobiles

The goal of GATES is to determine at any moment which behaviour each mobile of the simulation has to adopt. This behaviour is based on a set of plans, each one corresponding to a conditional sequence of elementary actions. Therefore, the purpose for each sensor/effector mobile is to determine autonomously and in real time which plan to apply.

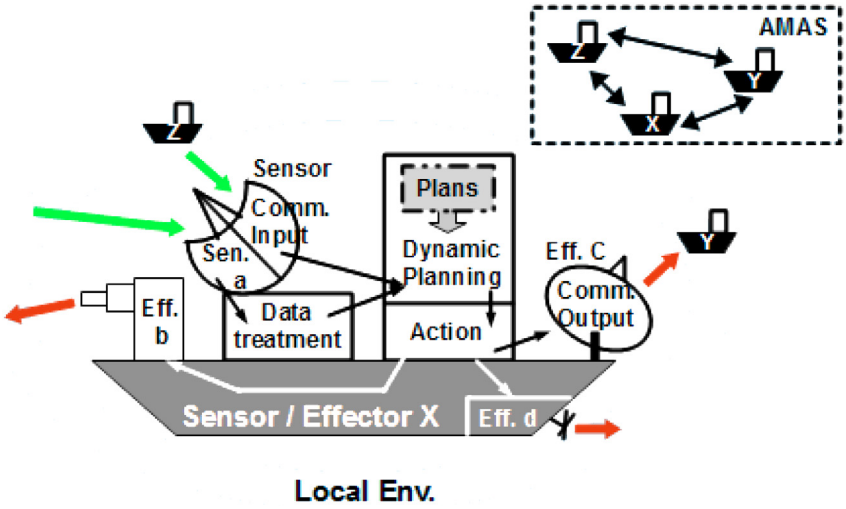

FIGURE 3. Mobile-agents interactions.

The first sub-part describes the general architecture of the adaptive multi-agent system while the emerging problem solving is presented in the second one. The request/proposal system we have developed is the subject of the last.

3.1. GATES ARCHITECTURE

In the early 2000s, the Institute of Computer Science Research of Toulouse (IRIT) has developed ADELFE, a suitable methodology for the design of AMAS. Going by this, we split up the application into autonomous agents each one representing a sensor/effector mobile of the simulation (see Fig. 3). To ensure its autonomy, each agent has different modules related to its capabilities.

A mobile-agent perceives its environment with a set of sensors such as the radar (sensor a in Fig. 3) or speed detector for example, and acts on this environment by the way of effectors such as weapons (effector b) or antennas (effector c) or movement controller (effector d). In particular the mobile communicates data with its allies.

During its lifecycle, all the information is processed into the decision module. This module is able to dynamically manage plans in order to adapt in real-time the mobile-agent behaviour according to the local situation.

There are two types of actions:

(1) those implemented by the most relevant plan using the mobile effectors;
(2) the communication to allies of all information and deductions which are deemed useful (*e.g.*: tell an ally that the mobile is willing to help it).

The actions of the agent affect the environment and the next perception-decision-action cycle starts.

3.2. From local processing to global problem solving

The plans management made by the decision module consists in determining the criticality of each plan-agent. At each lifecycle, this calculation is based on elements pre-determined or evaluated in real-time such as plan relevance towards mobile purpose, mobile ability to apply the plan and the current context.

$$\text{Plan criticality} = f(\text{relevance/aim, efficiency, context}).$$

From these data, a mobile-agent is able to rank its plans and to determine its own criticality. This latter is the one of the most critical plan.

$$\text{Agent criticality} = \max(\text{Plan}_i \text{ criticality}).$$

At this point, each mobile locally knows its criticality and the most relevant plan to reduce it. However, as this agent is a cooperative entity of an AMAS, its decision must take into account the global system interest before its own.

$$\text{System criticality} = \max(\text{Agent}_j \text{ criticality}).$$

The relevant behaviour of a cooperative mobile is to try to reduce the criticality of its neighbourhood.

(1) If the mobile is not able to apply its most critical plan, its own criticality will not decrease. As it represents a potential system maximum criticality, it informs the neighbours to complete their environmental knowledge. To do that, the mobile sends a message to its allies, which contains the plan that failed and its associated criticality.
(2) Each agent must take into account all messages received to update its knowledge about the collective.
(3) During the following decision phase, an agent determines the relevance of its plans in order to tackle the received problems of its neighbours.
(4) Autonomously, the agent establish then a new ranking considering the neighbourhood state based on its local perception (its intrinsic criticalities and those from the received messages).

This cooperative behaviour ensures that the most critical plan that failed in the system becomes the priority of all other agents. Therefore, if, at least, one mobile is relevant to tackle it, this ensures the criticality reduction of the system enabling the convergence towards a near-optimal solution. Moreover, we can notice two consequences of this approach. First, if a mobile is not the most relevant for the most critical plan, it will take into account the second and so on. This enables the AMAS to use its optimal local capabilities to address at a time the set of critical plans corresponding to the biggest criticality total. In addition, if the global criticality does not decrease, that means there is no solution.

We can also point out that the global resolution emerges from the mutual aid between mobiles. Furthermore, this help between mobiles is only a consequence

PLANS \ REQUEST	THREAT	SHIPWRECK	SUPPLYING	EVACUATION	NEURALIZATION	ESCORT
EVACUATE_ZONE				100		
REACH_ZONE				-100		
REACH_MOBILE	50	75			-100	50
RECEIVE_SHIPWRECKED		100	-100			
NEUTRALIZE_TARGET	80				100	
SUPPLY_MOBILE			100			
ESCORT_MOBILE	80				60	100

FIGURE 4. Example of Plans-Matrix.

of the agents' will; on the one hand to share information and on the other hand to reduce the global system criticality as perceived locally. Finally, we show that the global problem solving emerges from the individual behaviour of autonomous cooperative entities.

3.3. REQUEST/PROPOSAL SYSTEM

The presented design needs a system allowing each mobile-agent to ask for help and to offer its services to those in need. For this, new tools were introduced.

For a given request, several allied actions may fit. Conversely, for a given problem, the mobile must determine the most relevant request to send. Therefore, we have created the Plans-Matrix and the Requests-Matrix.

Figure 4 shows a short example of a Plans-Matrix (Pl-M) that allows the simulation leader to establish the (non-)relevance of each plan to respond to a given request. With these values, the decision module is able to modulate the criticality assigned to each plan.

If the mobile wishes to ask for help in order to realize a task, the Requests-Matrix (Rq-M) allows it to determine which request suits the best to get help regarding the most critical plan the agent wants to run.

The request/proposal system cycle is as follows:

– Update of other mobiles representation
  - The mobile removes each one of its non-selected proposals (by others).
  - For each waiting request from others for which the agent has a relevant plan (determined with Pl-M), it incorporates this latter into its decision module.

  - Among its selected proposals, using the coordination algorithm, the mobile determines the one for which the cancellation will be the cheapest. Then the agent incorporates it into its decision module and all other proposals are cancelled.

– Publication of proposals
  - The mobile publishes all new proposals it is able to perform and associates a cost with each one.

– Next action decision
  - If the selected proposal can not be taken on, the mobile cancels it.

– Publication of requests
  - If the mobile is failing, he publishes the appropriate request (determined thanks to Rq-M) or updates it if it is already published. The request is associated with the relative plan and its criticality. If the request already emitted is no longer useful, it is canceled.

– Proposals selection for the request sent by the agent
  - For the current non-satisfied request, the mobile selects the best proposal among those made by its allies. The request then becomes satisfied.

4. Theoretical support

Going by the ADELFE methodology, we obtain a system architecture perfectly compliant with the Adaptive Multi-Agent Systems theory. In this part, we first present a theoretical and methodological background for AMAS and then we provide the criteria for self-organization among agents of our system.

4.1. Theoretical and methodological background

4.1.1. *The Theorem of Functional Adequacy*

The IRIT has developed the AMAS theory [7] which is based upon a theorem that describes the relation between cooperation in a system and its resulting functional adequacy[1].

Theorem 4.1. *For any functionally adequate system, there is at least a cooperative internal medium system that fulfils an equivalent function in the same environment.*

Definition 4.2. A cooperative internal medium system is a system where no non-cooperative situations exist.

[1] "Functional" refers to the "function" the system is producing, in a broad meaning, *i.e.* what the system is doing, what an observer would qualify as the behaviour of a system. And "adequate" simply means that the system is doing the "right" thing, judged by an observer or the environment. So "functional adequacy" can be seen as "having the appropriate behaviour for the task".

Definition 4.3. An agent is in a Non-Cooperative Situation (NCS) when: (1) a perceived signal coming from the environment is not understood or is ambiguous; (2) perceived information does not produce any activity of the agent; (3) the conclusions are not useful to others.

The cooperation failures are called "Non-Cooperative Situations" (NCS) and can be assimilated to "exceptions" in traditional programming. The definition of cooperation is based on three local metarules the designer has to instantiate according to the problem to solve:

Metarule 1. *Every signal perceived by an agent must be understood without ambiguity.*

Metarule 2. *Information coming from its perceptions has to be useful to its reasoning.*

Metarule 3. *This reasoning must lead the agent to make actions which have to be useful for other agents and the environment.*

The theorem of functional adequacy means that we only have to use (and hence understand) a subset of particular systems (those with cooperative internal mediums) in order to obtain a functionally adequate system in a given environment [6].

4.1.2. *The Engine for self-organization*

The designer provides agents with local criterion to discern between cooperative and non-cooperative situations. The cooperative attitude between agents constitutes the engine of self-organization. The agents have to try to choose the most cooperative action when they can and also, to detect and to solve NCS when they occur. Depending on the real-time interactions the multi-agent system has with its environment, the organization between its agents emerges and constitutes an answer to the difficulties of complex systems modelling (indeed, there is no global control of the system) [4]. In principle, the emerging purpose of a system is not recognizable by the system itself, its only criterion must be of strictly local nature (relative to the activity of the parts which make it up). In that respect, the AMAS theory aims at being a theory of emergence.

4.1.3. *Engineering adaptive multi-agent systems: ADELFE*

ADELFE[2] [3] is a methodology devoted to software engineering of adaptive multi-agent according to the AMAS approach. ADELFE enables the development of software with emergent functionality and consists of a notation based on UML (unified modelling language) and AUML (Agent-UML) [2], a design process based on the RUP (rational unified process), a platform made up of a graphical design tool called OpenTool and a library of components that can be used to make the application development easier.

[2]ADELFE is a French acronym for "Atelier de Développement de Logiciels à Fonctionnalité Emergente". www.irit.fr/ADELFE

The design process covers all the phases of a classical software design (from the requirements to the deployment) in adding some specific steps to design adaptive systems. OMGs SPEM (Software Process Engineering Metamodel) has been used to express the ADELFE process and the SPEM vocabulary is used to expound the methodology: WorkDefinitions (WD_i), Activities (A_j) and Steps (S_k).

4.2. AMAS self-organisation

The theory at the root of the technology we use shows that the functional adequacy of an AMAS can be guaranteed only if agents are cooperative with each other. This requires to identify all non-cooperative situations (NCS) an agent may encounter during its autonomous evolution and to determine the corrective action to implement. In the case of GATES, it is clear that cooperation lies in the exchange of messages allowing mobiles to communicate their criticality and to help each other. NCS therefore concern the request/proposal system.

In the system we present, the cooperation between mobile-agents ensures the relevance of the exchanges. This implies the shared information is perfectly correct, so totally workable and useful.

Here are the NCS we identify:

Intrinsic incompetence: the agent is unable to apply its most critical plan.

Action: The agent sends a request for some help.

Useless query: the agent has asked for help but its need has disappeared.

Action: The agent cancels its request.

Useless proposal: the agent has offered to help but can not take on.

Action: The agent cancels its proposal.

Help concurrency: two agents want to help the same mobile.

Action: The requiring agent chooses the one for which the action will be the cheapest.

Selected proposals concurrency: the agent has offered assistance to two different mobiles and its two proposals were selected.

Action: the agent uses a coordination algorithm to select one proposal and cancel the other.

Unproductive help: a requiring mobile has selected a proposal from the agent but this latter is ultimately unable to offer an effective assistance.

Action: the agent cancels its help proposal.

Help incompetence: the agent receives a request for which it has no competence.

Action: the agent ignores the request.

As NCS of the studied AMAS concern the request/proposal system, our approach seems slightly similar to the contract net protocol (CNP) because a ship sends a request when needed and it chooses the most relevant helper within the response set. CNP is known to be best suited to problems decomposable into a set of quasi-independent subtasks with little need for global information or synchronization. Unfortunately it is not the case in GATES and consequently CNP falls into very sub-optimal tasks allocation; this is mainly why it is never used for distributed

constraint satisfaction and optimisation problems (DCSOP). In this regard, our approach might be identified as a solver for DCSOP.

We have also shown that the lifecycle of an agent requires explicit representation of goals, plans cancelation or dynamic planning. These aspects are known limitations of usual Belief-Desire-Intentions software agent models. Moreover a fleet management is typically a multi-objective and multi-constraint problem and a BDI model is not an algorithm to tackle it.

5. Related works and prospects

5.1. Related works

5.1.1. *On Board Training systems*

Most of On board training systems (OBT) developed for navies is based on a computer generated force (CGF) which aims at scheduling a predefined scenario where each entity interacting in the theatre of operations is simulated. To face the inevitable mutation of the scenario due to real-time operator actions, several methods have been implemented, mainly oriented on scripts. They are triggered on main predictable operator actions and they introduce planed modifications in the predefined behaviour of the simulated entities.

For instance, the recent OBT provided by DCNS in the scope of French program HOZIRON (Forbin class frigate) implements a Discrete Event Model (DEM).

The DEM approach provides means to adapt the scheduling of the scenario in coherency with new introduced events derived from trainees or trainer inputs. However, limitation of the model shall be highlighted: the auto-adaptability of the system mainly driven by Conditional class events is limited to some predictable and easy use cases (*e.g.* munitions firing event, ship damage event, meteorological parameters modification). Extending the model for processing evolved predictable actions does not seem to be relevant in comparison with the high level of complexity of the data management that this should introduce.

5.1.2. *Emergent problem solving*

The underlying reasoning of AMAS agents is based on cooperation. Cooperation was quoted by Heylighen as a relevant process: "Everybody will agree that cooperation is in general advantageous for the group of cooperators as a whole, even though it may curb some individuals freedom" [8].

Even if several authors have studied cooperation in computer science such as Axelrod [1] or Huberman [9], to our knowledge the AMAS theory is the single one used for emergent problem solving.

Emergent solving is very useful when the designer is unable to predefine precisely what the goal will be in complex dynamic problems, such as in GATES. We already used it in very different domains such as dynamic ontology management, dynamic ecosystem equilibration or flood forecast in STAFF project.

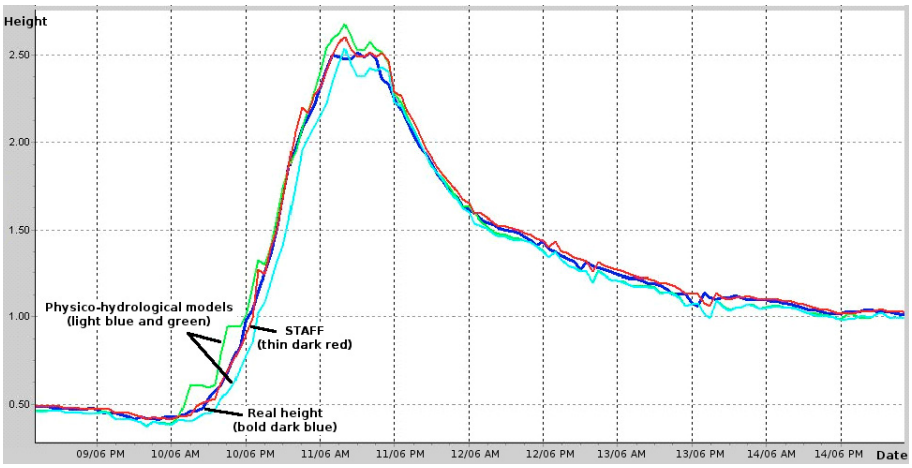

FIGURE 5. A five days flood forecast at Saint Girons station.

5.1.3. *Huge number of sensors*

The aim of the STAFF (software tool for adaptive flood forecast) software [5] is to be able to compute a flood forecast at any point in the Garonne basin without any prior information (either physical or hydrological) or specific configuration. It only utilizes the data from a lot of gauges in the basin and the real current river level at the point for which STAFF must provide a flood forecast. It must give such forecasts in real-time.

The result presented in Figure 5 compares STAFF forecast (3 h ahead) with two physico-hydrological models (2 h ahead only) specifically tuned for this river and the real height record. It clearly shows the STAFFs curve (thin dark red) is situated between the physico-hydrological curves (light blue and green) and so fits the best with the real situation (bold dark blue) although it forecasts an hour earlier. From evidence, STAFF is closed to the real river dynamic, whereas the two classical models, specifically tuned, are not so well. This is the general observation which could be done on the dozen of places where classical models are installed in Midi-Pyrénées. It highlights that our approach is perfectly adapted to integrate a huge number of sensors and to make them "smart".

We encompassed each sensor with an agent that is in charge of determining its influence. Each entry (typically one thousand of them) comes from sensors in the Garonne hydrological basin. At a station, all the sensors intervene as a pondered sum of their weights in order to compute the change in water level. Thanks to a systematic adjustment, the modelling is adaptive and the non-linear relation over time in spite of using a simple balanced sum, since the weights will change over that time. The aim of a sensor-agent is to adjust the extent to which the measure it is associated with affects the forecast.

Since 2002, STAFF has been operational on the Sophie software platform in France. STAFFs results indicate its real adaptation capacity based on a process of self-organization distributed among smart sensors. STAFF and the other applications using the AMAS theory are example of collaborative emergence, in which the goal is not to obtain a given end state but a never ending adaptation process because the systems are plunged into a dynamic environment.

5.2. Prospects

5.2.1. *A more evolved individual behaviour*

It appears that the use of emergence technologies allows governing coherently a set of sensor/effector entities influenced by unpredictable disturbances. By making the simulation mobiles autonomous and allowing them to interact, they are able to respond to unforeseen situations. The system becomes self-adaptive and dynamic.

However, a limitation appears concerning the individual scheduling. At every cycle, an agent has to determine its most relevant plan regarding to its own purposes and what its allies request.

Beyond the inter-mobile collaboration approach, it now seems necessary to explore the benefit of intra-mobile collaboration. Its consists in taking into account the impact each plan may have on other plans into the same sensor/effector. This would enable the execution of a non-priority plan for the purpose to make the realization of a more important plan possible.

GATES plans to design a second level AMAS where the agents will be introduced at the plan level. Mobile decision-making should take benefit from such an approach and then should enable to develop dynamic strategies.

5.2.2. *New generation sensors*

The presentation of the GATES project results shows how much the emergence technologies can be relevant to offer new abilities to a set of sensors. This subpart aims to explain that these new potentialities can be applied in more industrial cases.

By encompassing an AMAS module to the current sensors software, the sensors easily improve their autonomy whichl makes the system more dynamic towards the inevitable disturbances.

In industry, the most obvious and helpful example is provided by the information sharing seen above. For example, on a production line, each sensor all the time compares what it gets, first, to what it must perceive nominally (average of the last n, calibrated data), and then, to what it should perceive regarding to the information sent by other sensors. If a sensor disagrees with both during a period, it is able to declare itself in failure.

Furthermore, thanks to a better self-analysis based on its autonomy, a smart sensor self-applies basic corrective actions like the calibration procedure. If this does not correct its problem, looking forward to a reparation, the sensor will carry

on its task in degraded mode by self-correcting its perception regarding to all the information from its allies and its own knowledge.

6. Conclusion

This paper shows an innovative approach to handle multi-sensors systems in an adaptive and distributed way. Compliant with the Adaptive Multi-Agent Systems theory, such a design mainly relies on cooperation as the engine for coherent self-organisation of the system sensors. An implementation for Combat Management System training simulations is presented to demonstrate the relevance of this approach.

The agentification of simulation mobiles leads up to consider them as autonomous smart sensors/effectors. The examples of section 2 illustrate the emergent behaviours such a system can exhibit. The individual behaviour of each agent is based on a set of plans which are ranked according to two information sources. The mobile-sensor combines its own perception with the data communicated by its allies in order to determine locally the most relevant and compatible plans to execute in real-time.

Thanks to this design, each sensor/effector agent is able to self-adapt to all particular local situations and can also help its allies or asks for help. The request/proposal system used and the related lifecycle are detailed in the second part. For sensors integrated in complex systems with dynamic environment, it appears that the AMAS technology presented in Section 4 is a very relevant tool because the criticality of the agent collective is near optimal.

Many industrial sectors are now faced to mastering complex systems, designated by several terms in computer science: autonomic computing, pervasive computing, ubiquitous computing, emergent computation, ambient intelligence, amorphous computing... They possess many common characteristics:

- a huge amount of interacting elements;
- a variable number of elements during the system lifetime;
- the inability to impose a centralized control;
- an evolving and only partially predictable environment;
- a collective task to achieve which cannot be totally specified at the system design phase.

The complex system functioning depends heavily on their global structure and the intelligence degree of their elements (sensors or sub-systems). For complexity and distribution reasons, these elements cannot have the complete knowledge from the system environment and the other elements.

Consequently, self-organisation is required for the real-time adaptation of the global structure, as shown in this paper by the Amas technology. With the support of the suitable methodology ADELFE, we can easily foresee that this approach, nowadays mastered, will more and more penetrate the industrial world to offer new potentialities for dynamic complex systems incorporating numerous distributed sensors and effectors.

References

[1] R.M. Axelrod, *The evolution of cooperation*, Basic Books, New York (1984).
[2] B. Bauer, J.P. Müller and J. Odell, Agent uml: a formalism for specifying multiagent software systems, in *Proc. of the 1st international workshop, AOSE 2000 on Agent-oriented software engineering*, edited by P. Ciancarini and M. Wooldridge, Secaucus, NJ, USA, Springer-Verlag, New York, Inc. (2001), pp. 91–103.
[3] C. Bernon, M.-P. Gleizes, S. Peyruqueou and G. Picard, ADELFE, a methodology for adaptive multi-agent systems engineering, edited by P. Petta, R. Tolksdorf, F. Zambonelli and S. Ossowski, in *Proc. of the International Workshop on Engineering Societies in the Agents World (ESAW), Madrid, Spain, September 16–17, 2003*, number 2577 in LNAI, http://www.springerlink.com/ Springer-Verlag (2003), pp. 156–169.
[4] J.-P. Georgé, B. Edmonds and P. Glize, Making self-organising adaptive multiagent systems work, in *Methodologies and Software Engineering for Agent Systems*, edited by F. Bergenti, M.-P. Gleizes and F. Zombonelli, Kluwer, http://www.wkap.nl/ (2004), pp. 319–338.
[5] J.-P. Georgé, M.-P. Gleizes, P. Glize and C. Régis, Real-time simulation for flood forecast: an adaptive multi-agent system STAFF, in *Proc. of the Symposium on Adaptive Agents and Multi-Agent Systems (AISB), University of Wales, Aberystwyth, April 07–11, 2003*, http://www.aisb.org.uk, Society for the Study of Artificial Intelligence and the Simulation of Behaviour (2003), pp. 109–114.
[6] M.-P. Gleizes, V. Camps, J.-P. Georgé and D. Capera, Engineering systems which generate emergent functionalities, edited by D. Weyns, S. Brueckner and Y. Demazeau, in *Proc. of the Engineering Environment-Mediated Multiagent Systems – Satellite Conference held at The European Conference on Complex Systems (EEMMAS), Dresden, Germany, october 01–05, 2007*, number 5049 in Lecture Notes in Artificial Intelligence (LNAI), http://www.springerlink.com/ Springer-Verlag (2008).
[7] M.-P. Gleizes, V. Camps and P. Glize, A theory of emergent computation based on cooperative self-organization for adaptive artificial systems, in *Proc. of the 4th European Congress of Systems Science. Valencia Spain, September 20–24, 1999* (1999).
[8] F. Heylighen, Evolution, selfishness and cooperation; selfish memes and the evolution of cooperation. *J. Ideas* **2** (1992) 70–84.
[9] B.A. Huberman, *The performance of cooperative processes* (1991), pp. 38–47.