From Three MultiAgent Systems to One Decision Support System: the Representation MultiAgent System Layer

Patrick Person, Hadhoum Boukachour, Michel Coletta, Thierry Galinho and Frédéric Serin


Laboratoire d’Informatique du Havre (LIH)
25, rue Philippe Lebon,
Université du Havre,
76058, Le Havre Cedex, France

Abstract. This paper presents a system designed to help deciders manage cases of crisis. The system represents, characterises and interprets the dynamic evolution of a given situation and displays the results of its analysis. The core of the system is made of three multiagent systems (MAS): one MAS for the static and dynamic representation of the current situation, the second MAS for dynamically regrouping sets of agents of the former MAS and the upper MAS for matching results between the second MAS and scenarios stored in the persistent memory of the system in order to have a deeper analysis of the situation. The case based reasoning of this last MAS sends its results to the user as a view of the current situation linked to some views of similar situations. In this paper, we will focus on the representation MAS.

Key words: factual agent, ontology, semantic proximity, . . .

1 Introduction

This paper presents a global system designed to help deciders manage cases of crisis. The system could either be used to prevent a crisis or to deal with it. In both cases, the main internal aim of the system is to detect a crisis. From the system point of view, detecting a crisis implies representing a crisis, characterising a crisis and comparing a crisis with other crises permanently stored in scenarios. The result of this comparison is provided to the user as the answer of the global system. The system chooses to highlight parts of scenarios close to the current situation. We are confident that the information thus obtained will help deciders analyse the current crisis and its possible evolutions.

A genesis of our research precedes a general view of the architecture of the global system. Then the core of the system which is made of three multiagent systems (MAS) will be detailed. A common characteristic of these three MASs is the use of intelligent agents. Wooldridge and Jennings define these intelligent agents [20], [22] which are
the only kind of agents we will consider in this paper. Factual agents – which are our implementation for intelligent agents for the representation MAS – will be explained. In the fourth part, we will focus on the design and implementation in Java of this representation MAS. The game of Risk will be used as an illustration of the execution of this MAS. We will also use this game to precise which part of the MAS is generic and which one is specific, including ontology. We will then present the analysis of the choices we made about all the parameters and strategies we had to deal with. A presentation of the results and perspectives will conclude our paper.

2 Genesis

The story began with the work of Alain Cardon [8] i applying MAS to crisis management. Cardon postulated that the whole situation of crisis could be shaped into semantic features (SF) embedded inside what he called aspectual agents (AA). A SF is a three-part-relation \(<\text{object}, \text{qualification}, \text{value}>\) representing a partial aspect of the situation. A SF is also the atomic data structure, i.e. the smallest piece of information the system could deal with. An incoming data as a given SF could be transformed into a few AAs created to reflect different competitive supports of this particular SF: e.g. one agent could “agree” with the SF and another agent could “disagree” with it. Agents permanently compare their own points of view of the SF as a reaction to the evolution of the environment. Agents also try to reach their own aim. The aim of each agent is to reach a particular state inside its internal augmented transition network (ATN). Agent internal indicators can increase or decrease according to a proximity measure. The evolution of these indicators allows transitions inside the ATN. That way, the entire set of AAs can represent a situation.

From that work, Durand [10] and Lesage [14] developed a MAS in Smalltalk for the interpretation of speech during a crisis. This was the first implementation of AAs inside a MAS. The example chosen – interpretation of speech – allowed a numerical estimation of the proximity between two SFs and from there, between all the SFs and therefore between all aspectual agents: the more you talk about a topic or an aspect, the more important the topic or the aspect could be for the crisis. The proximity measure was coded in a proximity matrix. A second and last group of agents interpreted the activity of the AAs with a mix of their internal indicators. The analysis of the example proved the viability of the concepts.

Galinho et al. [13] and Vacher [19] used the same type of agents adapted to a job shop scheduling with additional abilities for the agents by the use of genetic algorithms and the use of a simple rule-based expert system for selecting strategies. An implementation was made in C++.

In the meantime, the analysis of Boukachour [3] was to split the representation and the interpretation of the situation into three MASs: one MAS for the representation, another MAS as an active observer of the representation MAS and the third MAS for comparing the observation with past observations. This architecture was tested on a scenario from an emergency exercise in an oil plant in Le Havre [2], [4]. An ontology of the specific domain was created to allow comparisons between SFs in this context [5].
The interested reader could download the four PhD theses mentioned above [15]. Although these works are in French, non-French readers could find useful references at the end of the theses.

Our team started an implementation of the three-layer architecture in Java [6]. We then carried out a cycle of software engineering which could be summarised as: dreaming of a system, building it, testing it. Tests on the representation MAS led us to redesigning agents. During the redesigning step, we chose to apply the system to the game of Risk [18]. We finished a second cycle of coding, fixing and testing the new system. The results of those tests provided us with meaningful information which we will develop later in this paper. We connected the representation MAS to the second layer that other members of the team was developing [7].

3 Architecture of the Decision Support System

The decision support system (DSS) is a tool whose main objective is to help deciders manage decision process in case of a crisis or before a crisis occurs. What this DSS offers to users is to analyse the current situation dynamically and compare it to past situations. The past situations are permanently stored in a scenario base and can be viewed as one part of the knowledge we have on the specific domain. Examples of the category of situations of crisis we are addressing are former works on interpretation of speech or emergency scenarios. Present work on the game of Risk could be set in the same category. Conjointly, we carried out works in the same category in other fields such as E-learning and scheduling actions in simulation of a crisis. In E-learning, the crisis with which the DSS must deal is spotting any problem a student could meet with during the learning process. The objective of the DSS is to be a “pedagogical agent” as presented in Bertin and Gravé [1]. As for scheduling actions, the DSS proposes an answer to deal with the crisis that occurs after an earthquake as proposed by the RoboCup Rescue team [17].

In order to be helpful for the decider, the analysis of the current situation must be of a great accuracy. We mean thus that the constarints the DSS must assume must be listed into details. Therefore the analysis must:

- present a synthetic view of the saillant aspects of the situation inaccordance with the role and personal interests of the given decider;
- present possible evolutions of the current situation with the associated consequences.
- respect a temporal constraint depending on the application. E-learning is the less time-constraint problem. But for other applications, the DSS must give accurate answers in a short span of time before some consequences happens. This last aspect adds a temporal constraint on the DSS: to react quickly enough according to the time scale of the problem.

Figure 1 shows the global architecture of this DSS. The inside query MAS and the inside information world are in charge of all the knowledge the core needs. The knowledge includes the scenario base we mentioned before. The knowledge also contains the ontology of the domain and the proximity measure which is specific to the domain. The
outside query MAS and the outside information world refer to the extraction and presentation to the core of the external information the latter could need and find in network distributed information systems. The presentation MAS will allow dialogue between all the users authorised to access the DSS and the core of the DSS. This MAS also presents users with the final results of the core.

Figure 2 shows the architecture of the core of the DSS. The environment provides a layer between the outer MASs presented in figure 1 and the three MASs of the core. The three internal MASs of the core communicate with each other and communicate with the environment. Each MAS has one and only one role. The representation MAS must reflect an accurate static view of the whole of the current situation and its dynamic evolutions. This MAS is built of factual agents (FA). As an example of the cardinality, we have about fifty FAs for the game of Risk. We had around two hundred FAs in previous emergency exercise scenario. The next section will detailed FAs. The characterisation MAS is an active observer of the representative MAS. The characterisation MAS clusters FAs thanks to the use of synthesis agents. Clustering regroups data sharing some common characteristics. Most of the time, clustering is applied to static data: you have the complete set of data – even if the data are imprecise, bias or uncertain – on which you use clustering methods. Here, the challenge is to deal with incoming data as soon as they arrive and consider the permanent evolutions of all the sets of FAs. Synthesis agents are an answer to this challenge: they cluster FAs both incrementally and dynamically. Clusters regroup FAs according to the evolution of their internal indicators. At any time, a new cluster could be created or an old cluster – which does not reflect the evolution any more – could disappear. The set of synthesis agents of the characterisation MAS is the internal view of the system, its internal representation of the current situa-
Fig. 2. Architecture of the Core of the Decision Support System

tion. The interpretation MAS takes that view, that observation and compares the current observation with past ones known as scenarios. The interpretation MAS is composed of prediction agents. They face the same challenge as synthesis agents: so prediction agents are incremental and dynamic. A prediction agent is associated to a given scenario or to a whole family of scenarios, depending on the applications. Prediction agents permanently try to match parts of their own scenario to the view of the current situation offered by synthesis agents. Through the environment, the activity of prediction agents is sent to the presentation MAS, and finally to users.

4 Intelligent Agents, Aspectual Agents and Factual Agents

Our MASs use intelligent agents. Quoting Wooldridge [22], intelligent agents must have the following capabilities:

**Reactivity.** Intelligent agents are able to perceive their environment, and respond in a timely fashion to changes that occur in order to satisfy their design objectives.

**Proactiveness.** Intelligent agents are able to exhibit goal-directed behaviour by taking the initiative in order to satisfy their design objectives.

**Social ability.** Intelligent agents are capable of interacting with other agents (...) in order to satisfy their design objectives.

Referring to a recent definition of aspectual agents (AA) given by Cardon [9], AAs are intelligent agents. Figure 3 gives the general structure of an AA [9].

---

1 pp. 23-25
2 figure 7. Structure générale d’un agent aspectuel, p. 93
Cardon proposes a knowledge module with an inference engine and a limited set of rules. For this module, we will however propose another possible approach later in this paper. Communications with other agents are basic ones. Behaviour – the module which leads actions or communications – is an augmented transition network (ATN). Actions will interfere with other agents and the environment.

Factual agents (FA) are aspectual agents for the general scheme. FA differs from previous AA realisations for the internal data representation, for the internal definition of the ATN, for the way they compute internal indicators and for the definition and the use of their internal acquaintances network.

The way FAs differ from AAs is the answer to a challenge we had to face: tuning the representation MAS. We had the MAS which we designed to observe the reactions of each agent and interactions between the agents. Experiments brought us the data we needed to tune agents and thereby the whole MAS. This led to a redesigning of the internal characteristics of our agents we have just listed above.

The second challenge we had to face was to identify the parts of the system which are generic and the parts of the system which are specific of a given application. We are not using generic with the same meaning as Wooldridge and Jennings [21] pointed out as a common pitfall of MASs. We use generic as the parts of the internal evolutions of an agent which is not directly influenced by a given application. The way we defined internal indicators will provide us with a first example of what we call generic. We have three internal variables: pseudoPosition, pseudoSpeed and pseudoAcceleration. The first one, pseudoPosition is a representation of the current position of the agent in the representation space. The evaluation of this position strongly depends on a particular application. However the computation of pseudoSpeed and pseudoAcceleration
does not depend on the application given the value of the pseudoPosition. To simplify
computation of the three variables, we use a constant value of 1 as a variation of time
between incoming data. This is why we refer to these variables with the prefix pseudo.

Talking about the internal data representation inside an FA, we will define a basic
semantic feature (SF) by its XML document type definition (DTD):

```
<!ELEMENT SF (KEY, QUALIFICATION, VALUE)>
<!ELEMENT KEY (#PCDATA)>
<!ELEMENT QUALIFICATION (#PCDATA)>
<!ELEMENT VALUE (#PCDATA)>
```

The DTD of a composite semantic feature (CSF) is:

```
<!ELEMENT CSF (KEY, (QUALIFICATION, VALUE)*)>
```

A CSF could associate a few couples of qualification-value to one object called key.
Inside the representation MAS, a given FA represents a CSF. As a CSF represents a
fact of the situation, the associated FA is the representation of this fact inside the MAS.
One CSF is associated with one FA. An evolution of the CST causes an evolution of the
associated FA. A new CSF provokes a creation of a new FA.

**Table 1.** Generic and Specific Characteristics of Factual Agents

<table>
<thead>
<tr>
<th>Module</th>
<th>Factual Agent</th>
<th>Generic</th>
<th>Specific</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knowledge</td>
<td>Composite Semantic Feature</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>. format</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>. content</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Behaviour</td>
<td>Internal Variables</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>. pseudoPosition</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>. pseudoSpeed</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td></td>
<td>. pseudoAcceleration</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Acquaintances Network</td>
<td>. definition</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td></td>
<td>. use</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Augmented Transition Network</td>
<td>. definition</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>. predicates</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>. action</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The specific ontology of the application is used by the proximity measure between
two CSFs. For every application, the proximity measure returns a value in [-1 .. 1]. A
value of -1 means a complete opposition between the two compared CSFs. A value of 0
means neutral or not comparable. A value of 1 means identity between the two CSFs and
any other value in this interval means a semantic connexion in the range from opposite
to identical. The proximity measure is multiplied by a coefficient specific to a given
application. From there, another example of a generic characteristic is the definition and
Table 2. Generic and Specific Characteristics of Representation MAS

<table>
<thead>
<tr>
<th>MAS</th>
<th>Generic</th>
<th>Specific</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proximity Measure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>. range [-1 .. 1]</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>. computation</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>. multiply coefficient</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>Ontology</td>
<td></td>
<td>x</td>
</tr>
</tbody>
</table>

the use of acquaintances network. The acquaintances network of each FA is a dynamic memory of agents whose proximity measure with the current agent is different from 0. Before we explain the use of acquaintances network, we have to present the internal augmented transition network (ATN) and the way we categorise incoming CSFs. An incoming CSF reaching the MAS is categorised according to specific criteria of the current application. That way we can use FAs and adapt them to adopt a few roles inside the representation system. From a generic five states ATN, we extract a sub-ATN according to the category of the CSF. At some predefined states of the ATN, we use acquaintances network to send flow of messages to request an increase or a decrease of the pseudoPosition of the receiving agent.

Table 1 summarises generic and specific characteristics at a factual agent level and table 2 at a representation MAS level. Referring to the previous aspectual agent architecture of figure 3, the knowledge part of an FA is made of composite semantic feature, the behaviour part is made of internal variables, an acquaintances network and an augmented transition network.

There are four kind of messages exchanged between FAs. Two of the messages are for passing information. The two other messages are for performing actions. Our messages are FIPA compliant [12]. Table 3 lists the messages with the FIPA performatives.

Table 3. FIPA Performatives of Messages exchanged between Factual Agents

<table>
<thead>
<tr>
<th>FIPA Performative</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>inform</td>
<td>IncomingCSFMessage</td>
</tr>
<tr>
<td>inform</td>
<td>TwoCSFMessage</td>
</tr>
<tr>
<td>request</td>
<td>SupportMessage</td>
</tr>
<tr>
<td>request</td>
<td>AgressionMessage</td>
</tr>
</tbody>
</table>

IncomingCSFMessage is a message with:

1. a composite semantic feature;
2. a categorisation of this CSF made by the system;
3. the time of arrival of the message inside the MAS.
This message can either be sent to a particular agent or to a group of agents or broadcast to all. A TwoCSFMessage is sent to a group of agents at least. This message contains two CSFs. The first one is used by the receiver to decide if the receiver is concerned by the second CSF.

There are three main states in the generic ATN which are Deliberation, Decision and Action. It is a generic behaviour for the ATN to send request messages when a transition from Deliberation to Decision occurs and when a transition from Decision to Action occurs.

Friends are agents which are close from a semantic point of view, i.e. agents whose compared proximity with the current agent is in $[0 .. 1]$. Enemies are agents which are opposite from a semantic point of view, i.e. agents whose proximity is in $[-1 .. 0]$. An FA sorts friends (enemies) from the strongest to the weakest, i.e. proximity from 1 to 0 (from -1 to 0). The two previous transitions in the ATN of an FA cause the given FA to send request messages to support friends and to agress enemies. Effects of different strategies of choices for friends and enemies will be discussed in the “results and perspectives” section. Receivers of request messages will freely decide to execute or not execute actions in consequence of the request message.

5 Design and Implementation of the Representation MAS

The representation MAS is mainly composed of FAs. The MAS use also an IncomingCSFAgent, a GenerativeAgent and an OutputAgent. To illustrate how the representation MAS works, we will use the game of Risk. The game of Risk is a board game with 42 territories and 6 players. A player wins by conquering all territories [18]. After an emergency exercise scenario, we choose a game to benefit from the following properties:

– the assumption of a closed-world;
– an easy evaluation of the quality of an advice given by the system (we know if the system helps us to win);
– a “reasonably” short time of execution thus allowing the system to loop and produce enough examples to test.

We choose the game of Risk because:

– it is not a toy problem;
– it is well suited for crisis management;
– we have an expert in our team.

From the original game, we keep only 42 territories and 6 players: we do not consider continents. Earlier, we saw that a CSF is categorised. Here, we have two categories: player and territory. We will have 6 FAs of category player and 42 of category territory. Figure 4 shows the generic ATN of an FA. Categorising FAs means:

– extracting a sub-ATN from generic one as shown by figures 5 and 6;
– determining predicates for each transition as shown by table 4. At any time, we have to have one and only one condition of transition true for the set of outside arrows of any given state of the ATN.
Fig. 4. Generic Augmented Transition Network of a Factual Agent

Fig. 5. Augmented Transition Network of a Player Factual Agent

Fig. 6. Augmented Transition Network of a Territory Factual Agent
Table 4. Predicates of Transition of the Augmented Transition Network of a Territory Factual Agent

<table>
<thead>
<tr>
<th>From</th>
<th>To</th>
<th>Predicate</th>
</tr>
</thead>
<tbody>
<tr>
<td>initial</td>
<td>initial</td>
<td>PP $&lt; 1$</td>
</tr>
<tr>
<td>initial</td>
<td>deliberation</td>
<td>PP $\geq 1$</td>
</tr>
<tr>
<td>deliberation</td>
<td>deliberation</td>
<td>PS $\leq 0$</td>
</tr>
<tr>
<td>deliberation</td>
<td>decision</td>
<td>PS $&gt; 0$</td>
</tr>
<tr>
<td>decision</td>
<td>deliberation</td>
<td>$(PS \leq 0)$ and $(PA \leq 0)$</td>
</tr>
<tr>
<td>decision</td>
<td>decision</td>
<td>$((PS &gt; 0)$ and $(PA \leq 0))$ or</td>
</tr>
<tr>
<td>decision</td>
<td>action</td>
<td>$(PS \leq 0)$ and $(PA &gt; 0))$</td>
</tr>
<tr>
<td>action</td>
<td>decision</td>
<td>$(PS \leq 0)$ or $((PS &gt; 0)$ and $(PA \leq 0)$</td>
</tr>
<tr>
<td>action</td>
<td>action</td>
<td>$(PS &gt; 0)$ and $(PA &gt; 0)$</td>
</tr>
</tbody>
</table>

To shorten notation in table 4, we will use PP for pseudoPosition, PS for pseudoSpeed and PA for pseudoAcceleration.

![Diagram](image)

Fig. 7. Internal Structure of the Representation MAS

Figure 7 shows an internal view of the representation MAS. The representation MAS contains FAs. The cardinality of the set of FAs depends on the application. For the game of Risk we have 48 FAs. The representation MAS also contains one IncomingCSFAgent, one GenerativeAgent and one OutputAgent. The IncomingCSFAgent is
activated as soon as a CSF from the environment reaches the MAS. The *IncomingCSFAgent* automatically detects the category of the CSF (player or territory) of the incoming CSF and stamps it with the time of arrival. Then, the *IncomingCSFAgent* send an *IncomingCSFMessage(CSF, category of CSF, time of arrival)* to the *GenerativeAgent*. When the *GenerativeAgent* receives that *IncomingCSFMessage*, it looks at the existent FAs of the MAS and decides either to create a new FA or to update a particular FA. A new FA is created if the *GenerativeAgent* considers that none of the existent FAs are in charge of this particular CSF. The “results and perspectives” section will propose a different strategy for this *GenerativeAgent*. The *OutputAgent* sends XML formatted output from FAs to its network connection. At the other end of the network connection, the receiving program uses these traces to dynamically monitors the evolution of the whole MAS, stores data in database for posteriori analysis and feeds the environment for the characterisation MAS.

An example of a CSF for a player is \(<\text{red}, \text{nbTerritories}, 0, \text{step}, 1>\) saying that player = red had 0 territory at step 1. An example of a CSF for a territory is \(<\text{Quebec}, \text{player}, \text{red}, \text{nbArmies}, 3, \text{step}, 7>\) saying that territory = Quebec have 3 armies belonging to red player at step 7. The first CSFs extract from the scenario will cause the creation of the 6 player FAs and then the 42 territory FAs. The following CSFs will cause the update of the 48 FAs. A CSF as \(<\text{red}, \text{nbTerritory}, 0, \text{step}, 1>\) will cause the creation of red player FA. A CSF as \(<\text{Quebec}, \text{player}, \text{red}, \text{nbArmies}, 3, \text{step}, 7>\) will cause the creation of territory Quebec FA. Creating a new territory FA causes a broadcast of a *TwoCSFMessage(CSF1, CSF2)* to the group of player FAs. The meaning of this message is: “as a new territory called Quebec, I belongs to red player”. So the red player FA will be able to update its number of territories. All the other players would receive the same message. We are here in situation where no information is hidden. When receiving a *TwoCSFMessage*, FA uses CSF1 to decide if it needs to update, and uses CSF2 as the updating information. Updating implies that there exists an FA in charge of the incoming CSF. *GenerativeAgent* propagates its *IncomingCSFMessage* to the given FA. The reception of the *IncomingCSFMessage* by the FA causes updating of it(s) internal CSF (knowledge) and its internal indicators (behaviour). The reception of this message also causes broadcasting the *IncomingCSFMessage* to all the other FAs. FAs could choose actions according to the semantic contained in its CSF: when a territory is updated, FA notices if the player who owns this territory has changed. In that case, the territory FA would also send updating *TwoCSFMessage* to the group of player FAs. As the FA in charge of the CSF just propagates the *IncomingCSFMessage*, all the other FAs – which are not in charge of this CSF and know it – will receive an *IncomingCSFMessage*. Then, the receiving agent uses the *proximity measure* to compare its own CSF and the received one and updates accordingly.

As an example, if the green player uses to own Alaska before the CSF \(<\text{Alaska}, \text{player}, \text{green}, \text{nbArmies}, 1, \text{step}, 49>\), the territory FA in charge of Alaska will update and every other FAs will compare this CSF with their own. If the purple player owned GB before the CSF \(<\text{GB}, \text{player}, \text{green}, \text{nbArmies}, 3, \text{step}, 50>\), the corresponding territory FA would also send messages to player FAs because of the change of owner of a territory.
There are two different ways of internal updating of FAs. The way the FA updates directly in case the agent is in charge of the incoming CSF, and the way other FAs perceive the CSF. An FA in charge of a CSF will start by updating its knowledge to reflect the current situation. The now old CSF will be replaced by the new one. We are dealing here with information which can be trusted: information is not uncertain nor imprecise. The next section will relax this constraint. The updating of player agents is quite straightforward as receiving message will carry a value of +1 or -1 to apply to the number of territories. That variation of value will then be applied to \textit{pseudoPosition} and consequently to \textit{pseudoSpeed} and \textit{pseudoAcceleration}. The new results of this computation selects the right predicate of the current state of the ATN. The chosen predicate could be a transition from state A to state A; the ATN remains in the same state. The predicate could also be a transition from state A to state B; the ATN changes state. We have a hierarchy of states of the generic ATN where \textit{Deliberation} $\prec$ \textit{Decision} $\prec$ \textit{Action} with \textit{Dead} as a final state. We progress in the ATN if we are transiting from \textit{Deliberation} to \textit{Decision} or from \textit{Decision} to \textit{Action}. We regress the way down. The behaviour of the generic ATN is to send \textit{request} messages when we progress in the ATN. This behaviour could be redefined for specific purposes. Variation that occurs into internal variables causes variation inside the ATN. As a result of the \textit{request} messages sent, the current FA could receive new messages.

Computation of internal variables respects the following formulae where \textit{PP} is a short notation for \textit{pseudoPosition}, \textit{PS} for \textit{pseudoSpeed} and \textit{PA} for \textit{pseudoAcceleration}:

\[
\begin{align*}
PP_{t+1} &= val \\
PS_{t+1} &= PP_{t+1} - PP_t \\
PA_{t+1} &= PS_{t+1} - PS_t 
\end{align*}
\]

Determination of \textit{val} depends on the category of FAs of a given application. These formulae explain why \textit{pseudoPosition} is specific and why \textit{pseudoSpeed} and \textit{pseudoAcceleration} are generic. The previous example is used again as an illustration of various way of computing \textit{val}. The situation before step 49 contains:

\begin{itemize}
  \item \textless Alaska, player, green, nbArmies, 3, step, 10\textgreater 
  \item \textless GB, player, purple, nbArmies, 4, step, 40\textgreater .
\end{itemize}

An incoming CSF: \textless Alaska, player, green, nbArmies, 1, step, 49\textgreater  is followed by CSF: \textless GB, player, green, nbArmies, 3, step, 50\textgreater .

In that sequence Alaska is only owned by green player. The computation of \textit{val} is given by: \textit{val} = new number of armies - old number of armies e.g. here \textit{val} = 1 - 3 = -2. GB is first owned by purple player and then by green player. The old number of armies refer to purple armies defeated by the new number of armies of green player. The computation of \textit{val} is given by: \textit{val} = new number of winning armies + old number of defeating armies e.g. here \textit{val} = 3 + 4 = 7.

The current implementation (AF1_0_6ER1) is developed in Java with Eclipse [11] and kernel and messages packages of MadKit [16]. Figure 8 shows the monitoring of the behaviour of FAs of the MAS. Each line is an FA. Each column, from left to right is a possible state of the ATN. The values of internal variables \textit{pseudoPosition},
Fig. 8. Monitoring of the Behaviour of Factual Agents

pseudoSpeed and pseudoAcceleration are given in the right part as a three-part-relation: PP, PS, PA. These values are also represented by the size of three coloured rectangles:

Fig. 9. Monitoring of the Knowledge of Factual Agents
red for pseudoPosition, green for pseudoSpeed and blue for pseudoAcceleration. The
column of these rectangles is the current state of the ATN. If a given agent reaches state
Decision at least once, the name of this agent is written in cyan colour. If a given agent
reaches state Action at least once, the name of this agent is written in magenta colour.
The bottom line shows the last updated values.

Figure 9 shows the monitoring of the knowledge of FAs of the MAS. Bottom lines
are for player FAs with colour and number of armies. The upper part is a visualisation
of the 42 territories with the colour of the player who possesses the territory and the
number of armies on this territory.

6 Results and Perspectives

Bayesian network
CATN : action d’un agent directement sur l’ATN d’un autre agent
Analyse des diverses strategies du reseau d’accontances
GenerativeAgent : comment creer/gerer pls agents pour un CSF incertain
information imprecise : comment prendre en compte

References
3. Boukachour, H.: Système de veille préventive pour la gestion de situations d’urgence: une
versity of Le Havre, 2002.
4. Boukachour, H., Simon, G., Coletta, M., Galinho, T., Person, P., Serin, F.: Preventive Mon-
toring Information System: a Model Using Agent Organizations SCI2002, Orlando, USA,
2002.
5. Boukachour, H., Galinho, T., Person, P., Serin, F.: Towards an Architecture for the Represen-
une architecture multiagent pour la représentation et l’évaluation de situations dynamiques,
CCGEI03, Montréal, 2003.
8. Cardon, A.: A multi-agent model for co-operative communications in crisis management sys-
tem: the act of communication. Proceedings of the 7th European-Japanese Conference on
9. Cardon, A.: Modéliser et concevoir une machine pensante : approche de la conscience artifi-
cielle, Vuibert, 2004 (?).
10. Durand, S.: Représentation des points de vues multiples dans une situation d’urgence: une
11. site de Eclipse
12. site de FIPA + ? num document(s)
shop scheduling. In Helder Coelho, editor, Progress in Artificial Intelligence, IBERAMIA’98,
16. site de MadKit
17. RoboCup 2005 ??
21. Wooldrige et ...