S-DLCAM: A Self-Design and Learning Cooperative Agent Model for Adaptive Multi-Agent Systems

Wafa Mefteh, Frédéric Migeon, Marie-Pierre Gleizes, Faiez Gargouri

S-DLCAM: A Self-Design and Learning Cooperative Agent Model for Adaptive Multi-Agent Systems

Wafa $MEFTEH^{*,**}$, Frédéric $MIGEON^{*}$, Marie-Pierre $GLEIZES^{*}$, and Faiez $GARGOURI^{**}$
**University of Paul Sabatier, IRIT, 118 Route de Narbonne, 31062 Toulouse, Cedex 9, France*
(mefteh, migeon, gleizes)@irit.fr
***University of Sfax, MIRACL, Pôle Technologique de Sfax BP 242, 3021, Sakiet Ezzit Sfax. TUNISIE.*
(faiez.gargouri@fsegs.rnu.tn), (wafa.mefteh07@gmail.com)

***Abstract*—Given the incomplete knowledge that an Adaptive Multi Agent System (AMAS) has on its dynamic environment, the detection and the correction of problems encountered called Non Cooperative Situations for the construction of the good behaviour of the AMAS agent can challenge even the most experienced designer. Our goal is to help the AMAS designer in his task by providing an agent behaviour able to self-design. In this paper, we propose a self-design and learning cooperative agent model.**

***Keywords*-Adaptive Multi-Agent Systems; Cooperative Agent; Self-Design and Learning Cooperative Agent Model.**

I. Introduction

For Adaptive Multi-Agent System (AMAS) [1], the development of adaptation implies the need to focus on the agent level. This is to give the agent the means to decide autonomously to change its relationships with other agents in order to move toward a cooperative organization. Thus, depending on the interactions that the AMAS has with its environment, the organization between its agents emerges. Building such self-organized systems is not a trivial task. In this paper, we propose a new cooperative agent model based on Self-Design and learning mechanisms developed from the agent model associated with the AMAS theory [1], [2], [3]. We take in account the following important works: [4] (in which Capera et al. present a model based upon a sort of extended automata product, dedicated to multi-agent systems) and [5] (in which Russel and Norvig present how an agent can find a sequence of actions that achieves its goals, when no single action will do). Indeed, we consider that the Self-Design and Learning Cooperative Agent (S-DLCA) life cycle goes through two levels: the preliminary level (PL) (nominal and cooperative behaviour) given by the designer and the heigh level (HL) which is responsible of the detection and correction of the Non Cooperative Situations (NCS) that the agent may encounter during its life. This model was developed under SeSAm (http://www.simsesam.de/) and it can be used by any AMAS designer in order to help him in the detection and correction of the NCS using the new ADELFE methodology extensions [6].

II. Redefinition and Location of the Non Cooperative Situations in the agent life cycle

We consider that the agent life-cycle goes throw three phases: *Perception & Interpretation (P & I)*, *Reasoning & Decision (R & D)* and *Communication & Action (C & A)*. We identify new types of Non Cooperative Situations that an AMAS agent may encounter and we locate them with the old ones in the agent life cycle (Table I).

P & I	• **Incomprehension:** the agent is unable to extract the information content of a signal; • **Ambiguity, uncertainty:** the agent is not sure about the interpretation that it assigns to a perception; • **Ambiguity, equivocation:** the agent gives two (or more) interpretations to the same perception.
R & D	• **Incompetence:** the agent is unable to exploit a given interpretation in its reasoning or decisions; • **Unproductiveness:** the use of a signal does not lead to any new conclusion. The information produced is either already known, either uninteresting or incomplete; • **Incoherence (Norms Violation):** this NCS can be encountered when the agent's reasoning leads to a conclusion that does not agree with its knowledge;
C & A	• **Inability:** this NCS can be encountered when the agent is enable to perform an action or a sequence of actions due to a **conflict** (its action is incompatible with the action of another agent), a **concurrence** (it make the same action as another agent), a **uselessness** (its action is useless for it and for other agents) or because the agent hasn't the skills that allow it to realize the action.

Table I
Redifinition and location of the NCS in the agent life-cycle

III. The static structure of the proposed agent model

Figure 1 presents the static view of our agent model and its different modules are defined in table II.

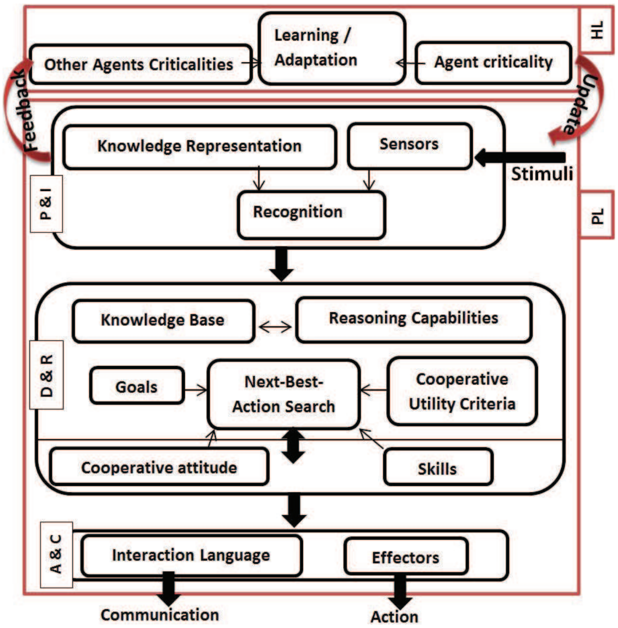

Figure 1. The static view of the Self-design and Learning Cooperative Agent.

Sensors	This module enables the agent to sense its environment.
Knowledges Representation	This module represents The language of representation used by the designer to represent the agent's knowledges.
Recognition	This module enables the agent to recognize/interpret what it perceives.
Knowledge Base	This module represents the agent's knowledge about itself and its environment.
Reasoning Capabilities	This module is given by the designer in order for the agent to reason on its interpretations using its knowledges.
Goals	This module represents the local goal that the agent must achieve in the form of a future agent state.
Skills	This module represents the atomic actions and the sequences of actions that the agent can perform.
Cooperative Utility Criteria	This module is responsible for the utility measurement of an agent state. It allows the agent to choose between different future states that can lead to its objective.
Next-Best-Action Search	This module is responsible for searching for the best next action to decide. The next best action is decided cooperatively.
Cooperative attitude	This module represents how the agent must behave to realize a good cooperation with other agents and its environment.
Interaction Language	This module represents the protocols used by the agent to interact with other agents.
Effectors	This module enables to the agent to realize a desired action.
Other Agents Criticalities	This module support the criticalities of the other agents. The agent must behave in the way that permits it to equilibrate the criticalities of the other agents.
Learning / Adaptation	This module is responsible for the learning of the agent in order for it to adapt to its environment and achieve the functional adequacy of the system.
Agent criticality	The agent must behave in the way that permits it to equilibrate its local criticality.

Table II
DEFINITION OF THE AGENT MODULES

IV. THE DYNAMIC BEHAVIOUR OF THE PROPOSED AGENT MODEL

The agent perceives its environment and stores its perceptions in the list of perceptions (L_p). It gives interpretations about these perceptions and stores them in the list of interpretations (L_i). It connects each perception *"p"* with the predicates *"pr"* it knows and likely to be perceived. $D_c(p,i)$ represents the degree of certainty of the interpretation given to this perception. Algorithm 1 presents how the agent can detect an NCS related to the first phase (NCS_p). Then, the

Algorithm 1 Detection of NCS_p

```
for all p ∈ L_p do
  Incomprehension(p) ← TRUE
  if L_i(p) IS NOT NULL then
    Incomprehension(p) ← FALSE
    if L_i = i then
      if ε < D_c(p,i) ≤ λ then
        Uncertainty(p,i) ← TRUE
      end if
    else
      Equivo(p) ← TRUE()
    end if
  end if
end for
```

agent reasons on the L_i using its Knowledge Base (K_b). It gives conclusions which are stored in the list of conclusions (L_c) and it decides what activities to perform and save them in the list of decisions (L_d) and finally schedules them and stores them in the list of scheduled activities (L_{sa}). Algorithm 2 illustrates the detection of the NCS related to the second phase (NCS_r). It is an **INCOMPETENCE** when an interpretation *"i"* is not used to produce any conclusion or to make any decision. It is an *Unproductiveness* when the

Algorithm 2 Detection of NCS_r

```
if L_c ⊂ K_b then
  Unproductiveness(i) ← TRUE
end if
if L_d = {} then
  Incompetence(L_c) ← TRUE
end if
```

given conclusion already exists in K_b. It is an *Incoherence* when the given conclusion breaks one or more agent's rules. Finally, it performs activities elaborated in the R & D phase and saves each performed action in the list of performed actions (L_{pa}). Algorithm 3 illustrates the detection of the NCS related to the third phase (NCS_a). A decided activity is not performed because there is a *Conflict* and/or a *Concurrence* and/or a *Uselessness* and/or the agent hasn't the skills that enable it to perform this activity. It is a *Conflict*

Algorithm 3 Detection of NCS_a

$Inability(a) \leftarrow FALSE$
$L_{npa} \leftarrow L_{sa} \setminus \{L_{pa}\}$
for all $a1 \in L_{npa}$ **do**
  $Inability(a1) \leftarrow TRUE$
  $Uselessness(a1) \leftarrow TRUE$
  $L_{rest} \leftarrow L_{sa} \setminus \{a1\}$
  for all $a_{rest} \in L_{rest}$ **do**
    if $(a1.ad \subset a_{rest}.pr)$ **then**
      $Uselessness(a1) \leftarrow FALSE$
    end if
  end for
  for all $Ag \in L_{pag}$ **do**
    $Cnflict(a1, Ag) \leftarrow FALSE$
    $Concurrence(a1, Ag) \leftarrow FALSE$
    for all $a2 \in L_{aAg}$ **do**
      if $(a1.ad = a2.ad) \wedge (a1.de = a2.de)$ **then**
        $Concurrence(a1, a2) \leftarrow TRUE$
      else if $(a1.de \subset a2.pr)$ **then**
        $Conflict(a1, a2) \leftarrow TRUE$
      else if $(a1.ad \subset a2.pr)$ **then**
        $Uselessness(a1) \leftarrow FALSE$
      end if
    end for
  end for
end for

if the deletions made by the execution of an action *(a1.de)* contain preconditions of a perceived agent's action *(a2.pc)*. The perceived agents are stored in the list of perceived agents (L_{pag}). The actions to be made by a perceived agent "Ag" are saved in the list of actions to be performed by a perceived agent (L_{aAg}). It is a *Concurrence* if all of the agent's action additions *(a1.ad)* (and respectively all deletions *(a1.de)* are among the additions *(a2.ad)* (respectively deletions *(a2.de)* of a perceived agent's actions. It is a *Uselessness* when the additions made by the agent's action *(a1.ad)* are not part of the preconditions of its other actions $L_{a1.pc}$ nor the preconditions of perceived agent's actions $L_{a2.pc}$.
To deal with the encountered NCS, the agent realizes a set of specific actions to go out from each type of NCS. For the *Conflict*, *Concurrence* and *Uselessness*, the agent can anticipate them if possible since the R & D phase. For each encountered NCS, the agent tries to follow the following actions:

- α: Relate the current situation to other previous situations based on its experience in order to find a way to correct the current situation.
- β: $if \rceil \alpha \Longrightarrow$ it tries to ask the perceived agents if they can help it.
- δ: $if \rceil \beta \Longrightarrow$ it asks the designer to improve his work by giving more examples (for an NCS_p), enhancing the reasoning capabilities and the knowledge base (for an NCS_r) or enhancing the skills (for an NCS_a)

We propose that the agent operates in two modes: the *"Experimentation Mode"* and the *"Deployment Mode"*. During the *"Experimentation Mode"*, when detecting a NCS, the agent can ask the designer if it can find a solution for it. When a correction of a NCS occurrence is proposed to the agent, it learns from this in order to avoid it the next time. After many executions, the agent should be able, during the *Deployment Mode*, to correct by itself the encountered NCS based on its learning from the many corrections made during the *"Experimentation Mode"*.

V. Conclusion

In this paper, we provide a new cooperative agent model for Adaptive Multi Agent Systems. This model is based on learning mechanisms to give the agent the ability to self-design. Our objective is to help the designer and facilitate his task by automating as much as possible the task of detection and correction of the Non Cooperative Situations.

References

[1] J.-P. Georgé, M. P. Gleizes, and P. Glize, "Conception de systèmes adaptatifs à fonctionnalité émergente: la théorie des AMAS," *Revue d'Intelligence Artificielle*, vol. 17, no. 4/2003, pp. 591–626, 2003.

[2] N. Daures, "Etude de la simulation de systèmes multi-agents pour la conception vivante d'agents dans la méthode ADELFE," Université Paul Sabatier, Toulouse, France, Rapport de Master Master Recherche IARCL, June 2006.

[3] S. Lemouzy, "Auto-ordonnancement coopératif de règles comportementales," Université Paul Sabatier, Toulouse, France, Rapport de Master, June 2007.

[4] D. Capera, J. Fanchon, J.-P. Georgé, and V. Camps, "A Generic Model Based on Automata for Multi-Agent Systems," in *European Workshop on Multi-Agent Systems (EUMAS), Brussels, Belgium, 06/12/2005-08/12/2005*. http://www.kvab.be: The Royal Flemish Academy of Belgium (KVAB), Dcember 2005, pp. 79–90.

[5] R. Stuart and N. Peter, *Artificial Intelligence - A Modern Approach (3. internat.ed.)*. Pearson Education, 2010.

[6] W. Mefteh, F. Migeon, M. P. Gleizes, and F. Gargouri, "Simulation Based Design for Adaptive Multi-Agent System: Extensions to the ADELFE methodology," in *WETICE 2013 Conference, 11th Adaptive Computing (and Agents) for Enhanced Collaboration (ACEC), Hammamet, Tunisie, 17/06/2013-20/06/2013*. IEEE.