

# ADELFE 3.0 Design, Building Adaptive Multi Agent Systems Based on Simulation a Case Study

Wafa Mefteh<sup>1,2(✉)</sup>, Frederic Migeon<sup>2</sup>, Marie-Pierre Gleizes<sup>2</sup>,  
and Faiez Gargouri<sup>1</sup>

<sup>1</sup> MIRACL Laboratory, University of Sfax, Sfax, Tunisia  
wafa.mefteh07@gmail.com, migeon@irit.fr, faiez.gargouri@fsegs.rnu.tn

<sup>2</sup> IRIT Laboratory, University of Paul Sabatier, Toulouse, France  
gleizes@irit.fr

**Abstract.** ADELFE is a methodology dedicated to applications characterized by openness and the need of the system adaptation to an environment. It was proposed to guide the designer during the building of an Adaptive Multi-Agent System (AMAS). Given that designing such systems is not an easy task, this leads us to provide means to bring the AMAS design to a higher stage of automation and confidence thanks to Simulation. ADELFE 3.0 is a new version of ADELFE based on a Simulation Based Design approach in order to assist the designer of AMAS and make his task less difficult. This paper focuses on a practical example of building an AMAS using ADELFE 3.0.

**Keywords:** Adaptive multi-agent systems · ADELFE methodology · Simulation · ADELFE3.0

## 1 Introduction

Some complex systems are qualified by the heterogeneity and diversity of actors involved, the large mass of data, by the distribution of the manipulated information and also by the dynamic of the environments in which they are immersed. Modeling such complex systems requires the use of efficient techniques. In recent years, several research works are interested in the development of new techniques best suited to this kind of problems. The AMAS (Adaptive Multi-Agent System) theory [1] [2] was proposed to model complex systems. This theory has shown that for a system immersed in a dynamic environment, a system in cooperative interactions with its environment is functionally adequate. The ADELFE methodology [4] [5] was proposed to guide the AMAS designers through an approach based on the RUP (Rational Unified Process). However, the AMAS theory stipulates that the designer must find all Non Cooperative Situations that an agent may encounter or create. For each of these situations, it must give the actions to be performed to ensure the agent to come back and stay in a cooperative state with others and itself. The problem lies in the definition of the

cooperative behaviour that is complete and accurate enough to give the agent any means to avoid what are called Non Cooperative Situations (NCS). Building such self-organizing systems is not an easy task. Our objective was to provide agents behaviours to self-design. The goal is to help the designer, to discharge him from the inherent difficulty in search of cooperative behaviour of agents and accelerate design time. For this, this we studied the contribution of simulation to design these systems. Simulation enables to improve and test the behaviour of an agent but also the behaviour of the collective. The ADELFE methodology was enriched by engineering processes and tools to achieve the "living design" that integrates modelling, programming and simulation techniques. For this, we dene three challenges in order to enrich ADELFE by engineering processes and tools to achieve the Simulation-Based Design; *Firstly*, we help the designer and discharge him from the difficulty of searching for the cooperative behaviour of agents (search for Non-Cooperative Situations, anticipating problems ...). Thus, we dene iterative activities that enable (i) the design of a preliminary version of the cooperative behaviour that (ii) is tested according to predefined situations and give information on what goes right and wrong in order (iii) for the designer to be able to analyze what is going wrong and to start a new design cycle of the cooperative behaviour. *Secondly*, we automate the design and provide tools that can determine what kinds of Non Cooperative Situations have been encountered by the agent and what kinds of cooperative behaviour cure the malfunctioning. *Finally*, we enable the designer to provide only a basic behaviour for the agent that could adapt to the Non Cooperative Situations encountered during runtime according to analysis and adaptation of its cooperative behaviour automatically. This "Living Design" of the agent, as we call it, defines the most efficient mechanisms of adaptation obtained so far. In [7], we presented in details the different contributions integrated in ADELFE3.0 and compared ADELFE3.0 to ADELFE2.0. The experimentation results [7] indicates that ADELFE 3.0 is better than ADELFE 2.0 for the all verified aspects. This paper gives an example of application of ADELFE3.0 in order to present how it helps the designer and make his task less difficult. The objective of this paper is also to serve as a guide for a designer who wants to use ADELFE3.0 with another case study. For this, we begin by presenting briefly ADELFE3.0 and more details can be found in [7]. Then, we present the case study the Red Donkey Game resolved using ADELFE3.0.

### 1.1 ADELFE 3.0

Initially ADELFE (figure 1) includes five phases that were initially inspired from the RUP and gathers twenty one activities, producing twenty six work products [5]. Each activity contains a set of steps which are carried out by different roles. Each step uses one or more input documents and produces one or more outputs. The different documents (inputs or outputs) have different types. ADELFE was described using the FIPA template [3]. ADELFE was proposed to guide AMAS designers. This design process takes into account the specific aspects of AMAS, including those relating to cooperation. With ADELFE3.0 [7] (figure 2),

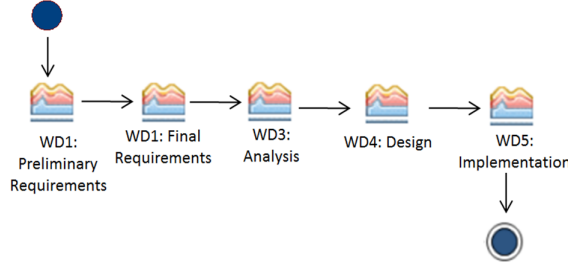


Fig. 1. ADELFE 2.0

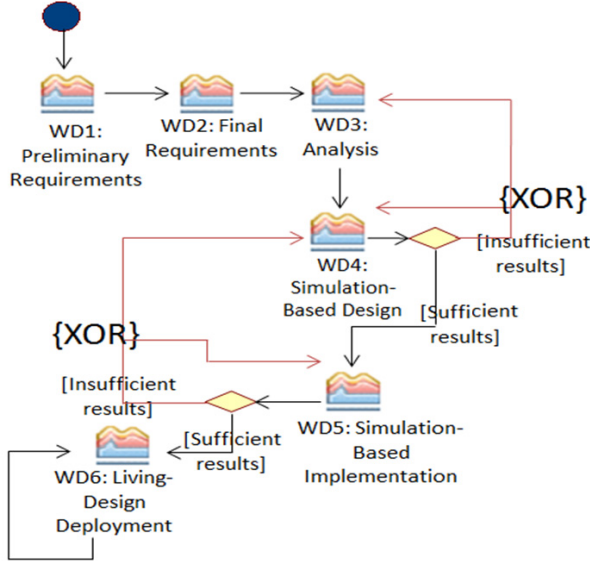


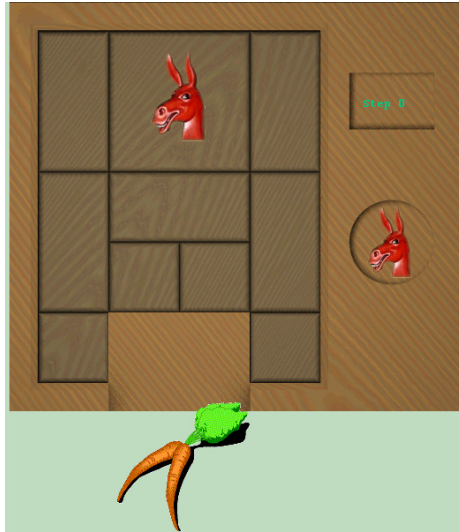
Fig. 2. ADELFE 3.0

we integrated a Simulation Based Design (SBD) approach. Our goal is twofold. *Firstly*, as usual, we want to shorten development time and lower costs by detecting as soon as possible misconceived parts of the system. *Secondly*, we would like to benefit as well as to enhance the adaptation capabilities of the Multi-Agent System by means of simulation-based techniques. So the ADELFE methodology was enriched by an engineering process and tools to achieve the Simulation-Based Design. Design, implementation and deployment phases are concerned. This issue includes techniques for modelling, programming and simulating. This technique of SBD permits to visualize information about the system that is running from models used in the specification of the system. With ADELFE, simulation helps in the design of the agent behaviour as well as in the validation activities and the construction of the self-design of the agents.

## 1.2 Case Study: The Red Donkey Game

The Red Donkey is a classical game originating from Poland where it is known as Klotski (or Klocki).

The original game is made of many wood blocks which can be arranged in various ways. The goal is to free the "master bloc" (which is the Red-Donkey) by successive shifts of the elements. The player is not allowed to remove blocks; he may only slide blocks horizontally and vertically. The objective is to exit the red donkey with a minimum number of moves. The classic Red-Donkey game structure is composed of 10 pieces. But our objective is to find a solution for this game considering a very large area with unlimited number of pieces. Only the number of empty cells and the pieces types are known.



## 1.3 Phase 1 (WD1): Preliminary Requirements

From a given initial distribution, the system must be able to exit the red donkey with the minimum possible moves and give the number of moves made to achieve the given solution. Considering that the number of the pieces is unknown, according to the user requirements, the following keywords are highlighted: *Board* (the ground in which the pieces move), *Cell* (a location in where a piece can be shifted), *Piece* (the object to move), *Move* (action of changing the pieces place without losing contact with the board).

## 1.4 Phase 2 (WD2): Final Requirements

We extract the following limits:

1. For a piece, a move is only possible if it does not lose contact with the board.
2. Minimize the number of moves.

We define three use cases as the following: *Set Initial Configuration* (the user sets the initial distribution of the pieces), *Calculate solution* (the user ask the system to calculate a solution), *Manage* (The user can stop, play, reset and restart the game). The user is the only actor interacting with the system. Based on the previous steps, the system can be designed as a multi-agent system. Indeed, the result given by the system is realized by the participation of all the entities of the system. Each entity is autonomous and has a well dened role.

### 1.5 Phase 3 (WD3): Analysis

The pieces are active entities because: *they are autonomous* (they decide themselves to move), *they have a local goal* (which is moving to a better place when it is possible), *they have a local vision about their environment* (each piece can perceive its neighbors (other pieces or free cells which are just behind it and also the door for the red donkey) and they can interact between themselves). The Cells (occupied and free) are passive entities because they cant change their state by themselves. Figure 3 presents the entities structure of the system. Each piece is able to move and check if this movement can cause a Non Cooperative Situation (NCS). The pieces, which are able to move, negotiate between themselves in order to let the most critical piece to move. After each move, a piece checks, with local knowledge, if the last move creates a problem or not. If yes, the problem must be corrected and this situation is saved in order to anticipate it next time. The interactions between entities are represented in figure 4. The next activity is to verify the AMAS adequacy. It permits the us to decide if the system can be built as an AMAS or not. ADELFE provides a decision tool which helps us to take this decision. Using the decision function dented in the AMAS adequacy tool to verify the Global AMAS adequacy, we obtain the result that our system can be developed as AMAS. Indeed, the global task of the system cannot be fully specified because though the result we want to achieve is known, the steps (moves) must be realized to get this result are unknown because of the dynamic of the environment. The solution requires a correlated activity of several components and several trials are required before giving an optimal solution. The environment evolves because it hasnt the same structure during the execution of the system. With each movement, pieces are facing a new environment with new empty places. A conceptual distribution is useful to solve the global task of the system. The system includes a considerable number of entities and it is *non-linear* because the entities are connected and the behaviour of an entity is influenced by the behaviour of the others. Active entities (pieces) are considered as agents because each piece takes its own decision (to move or not). A piece is able to change its state itself and moves when it is possible to a better place. Pieces interact between themselves and negotiate the best move. We define two

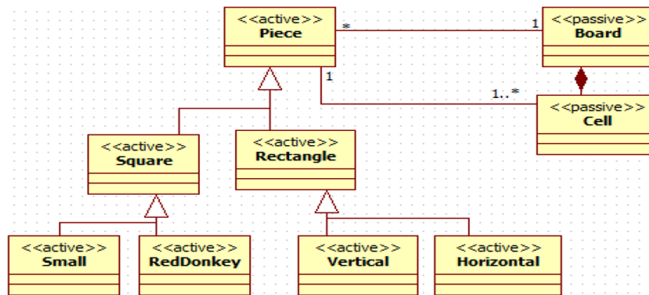
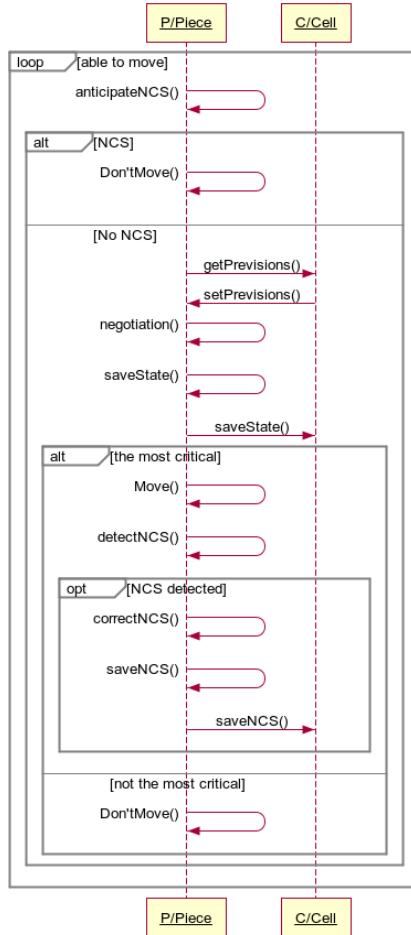


Fig. 3. Entities Structure Diagram

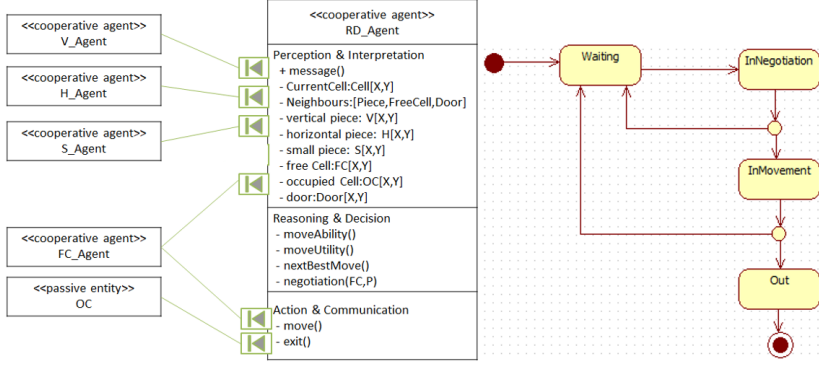


**Fig. 4.** Entities Interaction Diagram

cooperation failures between entities which are: *The USELESSNESS* if the move realized by a piece isn't useful neither for it nor for other pieces. Indeed, a useless move is a move that once it is made, the other pieces beside the free cells are blocked (they can't move) and the free place can not be occupied by another piece. *UNPRODUCTIVENESS* if the piece can not produce new information about the possible moves. All active entities must be cooperative agents and detect cooperation failures.

## 1.6 Phase 4 (WD4): Simulation Based Design

Figure 5 presents an example of piece agent (the red donkey) architecture and an inner state related to its behaviour. With the simulation tool SeSAM, we use

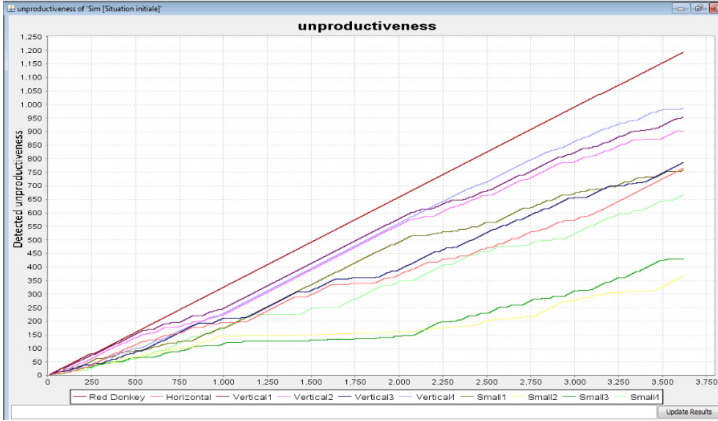


**Fig. 5.** (left) An example of a piece agent (the red donkey) architecture. (right) The inner state related to the behaviour of the `P_Agent`



**Fig. 6.** Detection of the USELESSNESS by each agent over time

the S-DLCAM model to develop four types of S-DLCAM [6] agents: Red Donkey, Vertical, Horizontal and Small. We give a preliminary behaviour (a non complete behaviour) to the agents. With Sesam, an agent behaviour is developed as an activity diagram which is composed of a set of activities and transitions between these activities. The S-DLCAM agent behaviour is composed of three types of activities: nomonal activities, NCS-detection activities and NCS-correction activities. To develop the agent of our system, we need to redefine some activities then, we execute the simulated system. The analysis of the experimentation results are given by Sesam in figure 6 and figure 7. These results indicate the number of NCS (USELESSNESS and UNPRODUCTIVENESS) detected by each agent during the same game. The horizontal axis presents the number of the step. The vertical axis presents the number of detected USELESSNESS (and UNPRODUCTIVENESS). The Red-Donkey agent detects a large number



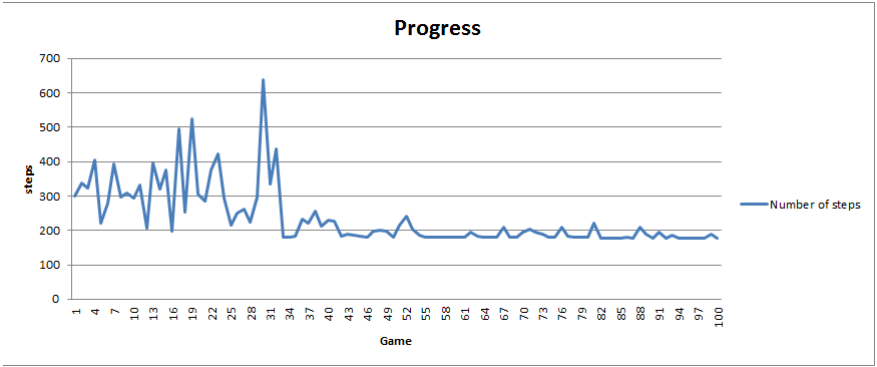
**Fig. 7.** Detection of the UNPRODUCTIVENESS by each agent over time

of USELESSNESS compared to the other agents but Small agents did not detect this type of NCS. This is explained by the number of cells which can be occupied by each piece (given its size). A Small can not block other pieces because when a Small moves, it occupies only one free cell and there is still another free cell which can be occupied by another piece (Horizontal, Vertical or Small). The curve of USELESSNESS of each agent is increasing in the beginning then it becomes constant which indicates that the agents detect less USELESSNESS over time. This is explained by the fact that the detected USELESSNESS have been corrected and so the agents do not fall in the same NCS next time or they have the ability to correct the problem if the detect a NCS detected before. The number of UNPRODUCTIVENESS detected by each agent is very large compared to the USELESSNESS. This NCS is generally more detected by the Red-Donkey than the other pieces. The Small detects generally less UNPRODUCTIVENESS compared to the others. This is explained by the fact that the more the piece is large the less it finds a possibility to move. The curves of UNPRODUCTIVENESS of all agents are increasing. Indeed, given that there are only two free cells so at maximum only two moves are possible at each step. We observe that all detected NCS are corrected automatically. According to these results, we the design can be validated because the agents are not blocked and they are able to overcome the problem when a NCS is detected.

### 1.7 Phase 5 (WD5): Simulation Based Implementation

The objective of this phase is to develop the system and also to verify that it is implemented exactly as described in the design phase. The system has been developed using the Jade platform. With JADE, the agent behaviour can be implemented as a set of activities and transitions (which is very suitable for AMAS agents because it allows us to focus principally on the behaviour of each agent). The execution of the system shows that it behaves exactly as expected in

the design simulations. The agents are able to detect the NCS and correct them. The system gives always a solution but the number of steps differs from a game to another. Figure 8 presents the progress of the system in terms of number of steps made by each game over time. The progress curve has a decreasing shape. This indicates that our system is improving based on its experience. It learns from the previous games (by detecting Non Cooperative Situations and correct them) in order to self-adapt and better play the next games (by avoiding the problems previously detected). The implementation is so validate and the system can be deployed.



**Fig. 8.** Progress of the system in terms of number of steps

### 1.8 Phase 6 (WD6): Living-Design Deployment

At this phase, we have only to install and activate the system. The system is able to adapt to its environment. The agents self-design during their lives by detecting, avoiding and correcting themselves the encountered problems (Non Cooperative situations) and the number of steps made by the system to exit the Red-Donkey from the board is fixed to the minimum.

## 2 Conclusion

Adaptive Multi Agent Systems (AMAS) are complex and self-organizing systems with emergent functionality. ADELFE was proposed to guide the designer of such systems. Building such systems is not an easy task because the designer has to exhaustively find all Non Cooperative Situations in order to obtain a coherent global behaviour. ADELFE3.0 integrates a Simulation Based Design approach in order to assist the designer in the detection and the correction of the Non Cooperative Situations that an agent may encounter during its life. This approach gives to the agent the ability to self-design. This paper gave an example of application of ADELFE3.0 [7].

## References

1. Gleizes, M.-P., Georg, J.-P., Glize, P.: Conception de systmes adaptatifs fonctionnalit mergente: la thorie des AMAS. *Revue dIntelligence Articielle* **17**(4/2003), 591–626 (2003)
2. Glize, P.: LAdaptation des Systmes Fonctionnalit mergente par Auto-Organisation Cooporative. Habilitation diriger des recherches, University of Paul Sabatier, Toulouse, France, June 2001
3. Cossentino, M.: Design Process Documentation Template. Experimental, 2011 IEEE Foundation for Intelligent Physical Agents, June 2011. <http://www.pa.org/>
4. Camps, V., Gleizes, M.-P., Bernon, C., Glize, P.: La conception de systmes multiagents adaptatifs: contraintes et spcificits. In: Association Francaise dIntelligence Articielle (AFIA). Atelier Methodologie et Environnements pour les Systmes Multi-Agents - Plate-forme AFIA, Grenoble, June 25–28, pp. 9–18, June 2001. <http://aa.lri.fr/>
5. Bonjean, N., Mefteh, W., Gleizes, M.-P., Maurel, C., Migeon, F. : ADELFE 2.0. chapitre/chapter dans. In: Cossentino, M., Hilaire, V., Molesini, A., Seidita, V. (eds.) *Handbook on Agent-Oriented Design Processes*, pp. 19–63. Springer (2014). doi:[10.1007/978-3-642-39975-6\\_3](https://doi.org/10.1007/978-3-642-39975-6_3), Print ISBN 978-3-642-39974-9, Online ISBN 978-3-642-39975-6
6. Mefteh, W., Migeon, F., Gleizes, M.-P., Gargouri, F.: S-DLCAM: a self-design and learning cooperative agent model for adaptive multi agent systems. In: WETICE 2013 Conference, 11th Adaptive Computing (and Agents) for Enhanced Collaboration (ACEC), Hammamet, Tunisia, June 17–20. IEEE (2013)
7. Mefteh, W., Migeon, F., Gleizes, M.-P., Gargouri, F. : Simulation Based Design for Adaptive Multi-Agent Systems with the ADELFE Methodology. *International Journal of Agent Technologies and Systems* **7**(1): 1–16 (2015)

Computational Collective Intelligence

7th International Conference, ICCCI 2015, Madrid,

Spain, September 21-23, 2015, Proceedings, Part I

Núñez, M.; Nguyen, N.T.; Camacho, D.; Trawiński, B.

(Eds.)

2015, XXVIII, 515 p. 140 illus. in color., Softcover

ISBN: 978-3-319-24068-8