Multi Agent System Based Approach for Industrial Process Simulation

Article *in* Journal EuropÃ©en des SystÃ¨mes AutomatisÃ©s · April 2021

DOI: 10.18280/jesa.540202

2 authors:

Lazhar Benoudina
Université 20 août 1955-Skikda
4 PUBLICATIONS **6** CITATIONS
SEE PROFILE

Redjimi Mohamed
Université 20 août 1955-Skikda
48 PUBLICATIONS **109** CITATIONS
SEE PROFILE

IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Journal Européen des Systèmes Automatisés
Vol. 54, No. 2, April, 2021, pp. 209-217
Journal homepage: http://iieta.org/journals/jesa

Multi Agent System Based Approach for Industrial Process Simulation

Lazhar Benoudina, Mohammed Redjimi*

Faculty of Science, Department of Computer Science, University 20 August 1955, LICUS Laboratory, Skikda 21000, Algeria

Corresponding Author Email: m.redjimi@univ-skikda.dz

https://doi.org/10.18280/jesa.540202

Received: 12 October 2020
Accepted: 25 January 2021

Keywords:
agent based modelling and simulation (ABMS), industrial system, modeling and simulation, agent/group/role (AGR) model, AALAADIN, MADKIT

ABSTRACT

Industrial systems become more and more complex. This complexity is due to the great number of elements that compose them and their interactions. This paper describes a multi-agent approach for modeling such systems. All of their parts are considered and are modeled by using adequate agents. The set of preoccupations were identified to find convenient multi agent models for their resolutions. Then, we implemented our application by using a MADKIT multi-agent platform. The main goal of this work is to build a simulator based on reactive agents able to translate this complex industrial system into a data processing programs that can represent its structure, its behavior, its interaction, its control loops and verify the integrity and its proper functioning. A concrete application of this approach was materialized by building an industrial gas process simulator.

1. INTRODUCTION

Modeling and simulation of industrial processes remain a major challenge. Many works were devoted to deep study this domain considering systems organization, matter and energy, collective behaviors and their inherent emergent properties, etc. Among the proposed modelling approaches, Agent Based Modelling and Simulation (ABMS) to decompose the global system on its constituents. The behavior of the complex system emerges from the local behaviors of the various interacting entities that compose it [1, 2]. In modern industry, advanced technologies are employed to develop industrial processes. The vast majority of these systems can be considered as complicated or complex in the sense that they are composed of a large number of interacting elements.

The concern of the work presented here deals with the modelling and the simulation of an industrial process for a gas plant. Our team has conducted various similar projects by using different approaches (DEVS, MAS, etc.) [3-5]. This project is part of the training of new recruits; so that they can follow and understand how the process works.

The use of computer tools for modeling offers the advantage of building a computer model that is very faithful to the considered system or to a set of its parts, then to judge the behavior of the obtained model as a function of the inputs. This makes it possible to handle, to observe and to improve understanding of its behavior and allows avoiding many possible errors concerning its future evolutions. The shutdown of certain large so-called critical systems can cause material and financial damage. For these systems, the use of models representing them is of substantial help. In fact, instead of stopping the operation of all or part of a plant, it suffices to consider the models, which represent them. This is the case, for example, with industrial boilers. The shutdown of such equipment can cause financial damages of the plant and to restart it may take a long time.

The modeling of systems has been the subject of many studies and researches considering its important impact on the phenomena implemented. Fishwick's [6, 7] considers the modeling as a set of metaphors and analogies to better understanding any phenomena [7], defined the computer simulation as follows: "Computer simulation is the discipline of designing a model of an actual or theoretical physical system, executing the model on a digital computer, and analyzing the execution output". Three fundamental phases are considering by this definition: The model design, the execution of the obtained model and the obtained results. Furthermore, Zeigler et al. [8], detail the simulation process according to the following criteria:

(1) The abstract or real source system of interest.

(2) The conditions of system experimentations.

(3) The system specification that defines the model: A computer program, a set of mathematical rules, equations or other constraints for the system behavior generation.

(4) The computation system so-called the simulator which is a system capable of executing the obtained model to generate its behavior.

(5) The modeling relation, which deals with the relation between a model, a system and an experimental frame. It defines the validity of the model, its ability to faithfully capture the system behavior within the extent demanded by the objective of the simulation study.

(6) The simulation relation, which defines the relation between a simulator and a model. It deals with the ability of the simulator to correctly simulate the model, to faithfully generate the model's output trajectory given its initial state and its input trajectory.

From these definitions, one can deducts that the model experimentation is what we say its simulation. Practically, a simulation is the execution phase of the model. Given a set of pertinent data as input of the model, this phase consists to make evolving the model and to observe and analyze the obtained results.

In some cases, simulation of complex systems can take hours or even days and consumes a lot of CPU time. This is why developers have recourse to methods of reduction of the complexity. Among these methods, one can cite parallel

simulation, the decomposition of the phenomenon in several entities of less complexity or the use of fast and parallel CPUs [9-11].

The search for high fidelity, precision and adaptation requires that modeling and simulation data-processing tools be chosen with great care. Their use in the field of modeling and simulation is very widespread ensuring excellent qualities for which the improvement is continuous with very reasonable costs.

Several modeling approaches, methods and tools have been and continue to be used in the literature.

Petri nets (PN) [12] are among the most efficient and widely used tools. These networks, despite their simplicity, make it possible to represent the dynamics of systems. Several variations and extensions of these networks have been developed such as Colored PN (CPN) and Timed PN [13, 14]. Despite their 62 years of existence (PN was introduced by Adam Petri in 1962 in his PhD. Thesis), they continue to be relevant today.

The Unified Modelling Language (UML) state chart diagrams are other powerful semi-formal tools that are able to represent the behavior and the interactions between the actors of a system [15]. Multi-agent systems (MAS) are used modeling tools that offer the advantage of a fine decomposition of a real complex system into a set of its components. The behavior of the global system emerges from that of the behaviors of its entities. The notion of agent designates an entity whose functioning is autonomous and which is capable of carrying out a set of tasks [16, 17]. These systems are used more and more in computer systems because of their flexibility, their simplicity, their effectiveness, and their possibility of extension. Moreover, the description given by a MAS can easier be understood by a human observer.

In this work, we adopted a top-down method for the industrial simulator design and used a Agent Based Modelling and Simulation (ABMS) approach for these aforementioned advantages. The whole system is decomposed in pertinent sub-systems. A reactive agent represents each of them. The communication between agents is ensured by message passing. To design the simulator model, we opted for the AALAADIN organizational model. We used the MADKIT (the Multi Agent Development KIT) multi-agent platform for the implementation of the obtained model. Both AALAADIN and MADKIT were developed at the Laboratoire d'Informatique de Robotique et de Micro electronique de Montpellier (LIRMM) [18-23] and use the Agent / Group / Role approach.

The rest of this paper is organized as follows: Section 2 is dedicated to related works. Section 3 presents the multi-agent systems based approach. Section 4 concerns the proposed approach. Section 5 introduces a use case of an industrial system simulation. and finally, section 6 concludes this paper.

2. LITERATURE REVIEW

The importance of the Multi Agent Based Simulation has attracted a number of research works concerning several application domains, which have been addressed in recent years. There are a large number of scientific articles dedicated to this subject.

The intuitive and simplistic idea of a multi-agent organization is to distribute the components of a global system into sub-sets of lesser complexity which can have autonomous operations and which can interact to achieve a common task.

Survey reported by Abar et al. [24] gives a precise idea of the scope of the fields covered by modeling and multi-agent simulation as well as the software tools dedicated to each field. Bandini et al. [25] describes the agent based modelling and simulation problems and some application domains.

Chen et al. [26] describe the modeling and the simulation of stochastic heat pump usage behavior in residential communities, which is a mechanical domain. Mahmood et al. [27] reports an agent-based modeling simulation of the natural phenomenon related to the rapid prototyping of wind power plants. Macal [28] present a multi-agent systems dedicated to modelling systems engineering and management problems. Herrera et al. [29] is a survey of the applications of the MAS in the systems engineering. Zhou et al. [30] present a survey of agent-based simulation packages that concerns the energy consumption markets. The paper investigated an agent-mediated simulation framework aiming to facilitate the development of models for electricity markets.

The field of education has also been invested in multi-agent modeling. A MAS based modelling and simulation of intelligent teaching system is presented [31].

Many contemporary contributions for agent-based modeling for the IoT and the cloud-fog computing have been surveyed [32, 33]. Nanna et al. [34] presents a Multi-Agent System for Simulating the Spread of a Contagious Disease.

3. THE MULTI AGENT SYSTEM

Multi-agent systems are an emerging conceptual paradigm to simulate the interaction of multiple autonomous agents in an environment [35-44]. Multi-agent systems have many applications; our interest here is in their use to build an operational simulator of an industrial system. In general, a system is called multi-agent when it contains a set of interacting agents, which evolve in some environments. An agent is influenced by the perceived situation in the environment and react on it. Multi-agent Systems have been in the research arena for at least 40 years now. To our knowledge, the foundations of MAS date back to around 1980 when these systems were introduced during the Workshop on Distributed Artificial Intelligence (DAI), held at the Massachusetts Institute of Technology (MIT) in June 1980. MAS was identified as a branch of the distributed artificial [45, 46].

There is no common agreement about a definition of the term agent. In general, we can say that an agent is an entity (hardware or software) which is different from an ordinary program in the sense that it is endowed with autonomy for functioning in environments shared by other agents [38, 42]. It is able of evolving in an environment through the acquisition of information emitted by the latter and by acting on it. The architecture of an agent varies from a simple structure formed of production rules often linked to very complex structures where the agent can be endowed with advanced mechanisms of cognition and reasoning referring to it an associative and social character [42].

Ferber [38] definition of the term 'multi-agent system' refers to a system consisting of the following parts: <O,A,L,R,Op,U>

where,

O, A and L constitute sets of the environment E.

O is a set of objects that can be manipulated by the agents (percepted, created, destroyed and modified).

A is a set of agents. The agents are the active entities of the system. They are able to perform actions.

L is a set of locations. It determines the possible positions of the agents in space.

R is a set of relations to link agents and objects.

Op is a set agent's operations.

U is a set of operators called universe law.

4. THE PROPOSED APPROACH

The work concerned by this present theme is related to the development of industrial process simulators in the hydrocarbon sector. This involves providing educational support to technical staff so that they can understand the different stages of the process.

The proposed approach is a top down approach, which starts, therefore, from the overall process and then divides it up according to the fundamental units (see Figure 1) after having identified them.

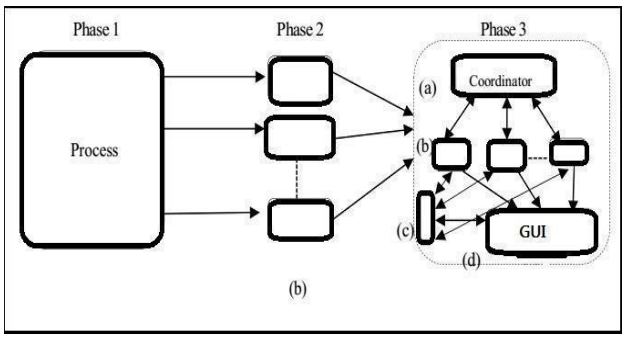

Figure 1. The decomposition process

The process cutting procedure is carried out as follows:

(1) In phase 1, the process is studied in detail. The principle of its operation is clearly established and all its the atomic components are identified. This phase concerns the general process investigations.

(2) In phase 2, the different units making up the process are identified according to the detailed technical diagram of the process (b). These units will constitute the agent groups that constitute the modeled system.

(3) In phase 3, the model of the simulator is developed each unit is isolated to be individually modeled and the obtained units are assembled in a global diagram.

Three additional control and communication units are added to the model:

(a) A coordinator whose role is to control and manage the execution of the whole system.

(c) An input interface through which the user can act on the system.

(d) An interface for visualizing the progress of the process according to the inputs sent by the user.

We have adopted this approach for the design of our simulator.

4.1 AALAADIN organizational model

For the design of this kind of simulators, we opted for a solution using multi-agent systems. An agent represents each unit. The same goes for the coordinator, the input interface as well as the display interface.

Our choice fell on the AGR (Agent-Group-Role) approach. This approach was developed at the computer science, robotics and mechanics laboratory in Montpellier (LIRMM) [18]. The AALAADIN meta-model is an organizational model for the modeling of multi-agent systems based on AGR concepts and it offers a set of conceptual tools for the implementation of applications based on multi-agent systems. The basic principles of this approach are briefly explained bellow and illustrated in Figure 2 [19, 21].

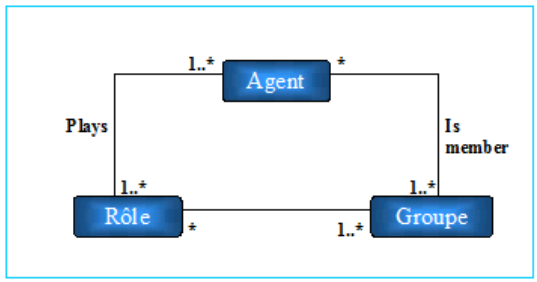

Figure 2. The AGR model

In the AGR model, an agent can play one or more roles; he can be a member of one (at one group) or more groups. A group is made up of at least one agent. An agent who does not belong to a group can request to join this group. Several agents can play the same role. An agent can communicate with another only the two agents are in the same group (Figure 3).

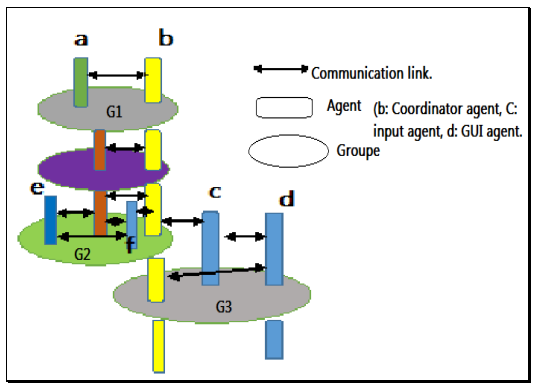

Figure 3. Example of organizational AGR model of the simulator

4.2 The MaDKit platform

The Multi Agent Development Kit platform is a free software for the design and the simulation of Multi-Agent Systems (MAS) [23, 47]. MadKit is a general purpose multiagent platform with agent based simulation layer under licence GPL (General Public License) and LGPL (Lesser General Public License). there are several online documentation, forum, FAQ and examples about this MAS platform.

The MadKit platform is built around the agent/group/role (AGR) concepts. There are three basic principles in this platform:

- Micro-kernel architecture
- Agentification of services
- Component model for graphical interface

MadKit is a set of packages of Java classes. It implements a reduced size agent kernel, various libraries of messages, probes and agents. In addition, Madkit includes a useful

graphical development environment and many other agent handling systems. Madkit implements different tools for the management of agents, groups and roles according to the AGR model. This platform is based on the AALAADIN meta-model.

5. USE CASE

Pure natural gas (NG) is a fuel unfit for consumption due to the presence of impurities, particularly water and carbon dioxide (H_2O and CO_2), which reduces its calorific value. Liquefaction of NG requires cooling to about -165°C. Knowing that the freezing temperature of the water is at 0°C and that of the CO_2 at -20°C, this will clog the pipes of the machines during the liquefaction process. It is for these reasons that NG must be treated (decarbonized and dehydrated) before the liquefaction operation. Decarbonation is the extraction of carbon dioxide CO_2 from natural gas and dehydration consists of extracting water from it. There are several industrial ways to execute the decarbonation. Presents the process adopted in this work [48].

5.1 The natural gas decarbonation process

We consider, here, two operations for the decarbonation process: i) the washing operation, which is intended to extract the CO_2 from the NG with an M.E.A ($C_2H_5ONH_2$), and the regeneration phase, where the results from the washing operation enters a regeneration tower so that they release CO_2 and then restore the MEA solution. Figures 4 and 5 illustrate the dedicated equipment.

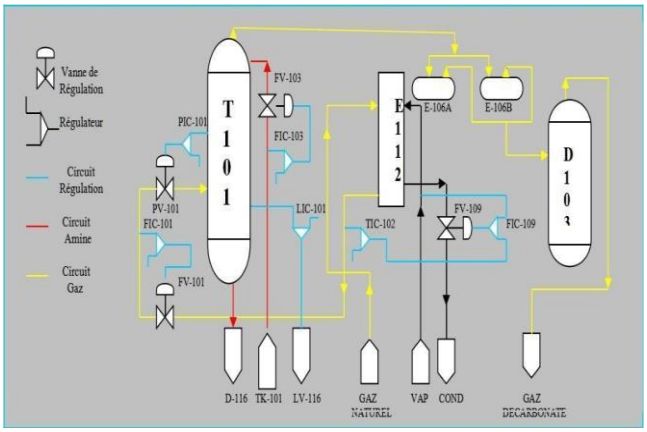

Figure 4. Technical drawing of the washing process

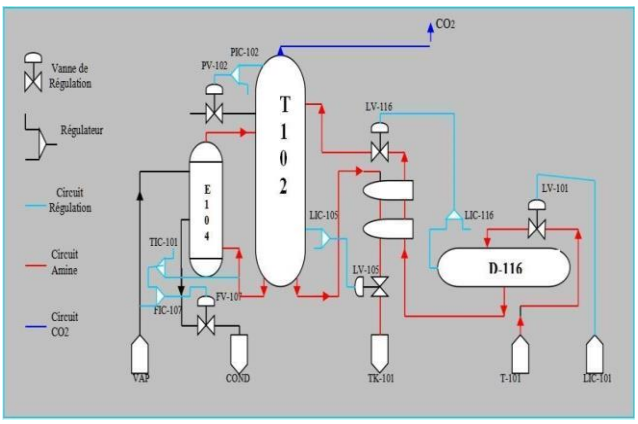

Figure 5. Technical drawing of the regeneration process

According to the breakdown system phases presented Figure 1, at first and after studying and analyzing the whole system as presented in Figures 4 and 5, the following four points are highlighted:

- **The system ordered:** it is the decarbonation process. It contains a set of organs (T-101, T-102, D-116, E-112, E-104 ...) which carry out the various chemical reactions to achieve the decarbonation of the Natural Gas.
- **The control system:** this is the distributed control system for the natural gas decarbonation process. It is formed by a set of autonomous, modular and dis- tributed instruments. They are intended for the control and regulation of the process. These instruments are divided into two sections:
- **The washing section:** it groups the regulation valves (FV-109, FV-103, FV-101, PV-101), which are regulated by a control system, which contains the different regulation loops.
- **The regeneration section:** also contains a set of control valves (FV-107, PV-102, LV-116, LV-105, LV-101), which are regulated by the same control system. To carry out this adjustment the control system is based on the set of instructions.

5.2 The proposed AGR model

To build an interactive simulator for the above-mentioned industrial gas process, the three following groups as depicted in Table 1 have been distinguished. Note that this point corresponds to the phase 2 of the proposed simulation approach. This decomposition was obtained after a meticulous study of the NG decarbonation process as well as the material structure and the arrangement of the interactions between the different material components (pressure, level, flow control, etc.). This article will not go into the precise details of how the system works.

The APM agent ensures the time synchronization between the different units. This particular agent (APM) serves as a coordinator in the system.

This decomposition was obtained after a meticulous study of the NG decarbonation process as well as the material structure and the arrangement of the interactions between the different material components (pressure, level, flow control, etc.). This article will not go into the precise details of how the system works.

The APM agent ensures the time synchronization between the different units. This particular agent (APM) serves as a coordinator in the system.

5.3 The AALAADIN model

The entire model with links between all the components (agents) of the system is depicted in Figure 6. This step corresponds to the phase 3 of the proposed demarche.

5.4 The system implementation

The simulator was developed by using the Mad Kit agent platform based on AALAADIN model.

The Mad-Kit architecture is based on the following three design principles:

- Micro-kernel architecture.
- Systematic agentification of services.
- Use of a componential graphics model.

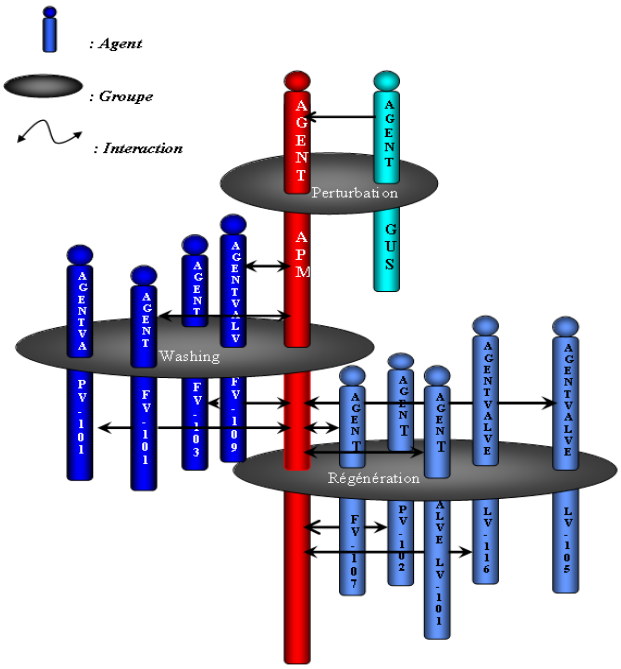

Figure 6. The AALAADIN model of the LNG simulator

Table 1. Obtained agents, groups and roles

Id_Group	Id_Agent	Role
Perturbation	GUS	To interact with the system
	APM	To acquire the Instructions
Washing	APM	To set and to regulate the washing group
	FV-103	To control the Amine flow
	FV-101	To check the natural gas flow
	PV-101	To check the pressure of T-101
	FV-109	To check the steam flow of E-102
Regeneration	APM	To set and to control the regeneration group
	FV-107	To check steam flow E-104
	PV-102	To check the pressure T-102
	LV-101	To check the level of T-101
	LV-116	To check the level of D-116
	LV-105	To check the level of T-102

Noto:
APM: The Agent Process Manager is the coordinator agent.
GUS: The Graphical User System. It provides the interface between the operator and the system
The other agents represent the functional units of the process:
FV: Flow Agents (Control of the gas flow)
PV: Pressure Agents (Control of the gas pressures)
LV: Level Agents (Control the of levels).

Mad-Kit's micro kernel is a small agent execution environment. It provides the following functionalities:

5.4.1 Management of local groups and roles

The micro kernel provides the basic Mad-Kit operations at the lowest level such as the management of the organizational structure (Agent / Group / Role), which allows the use of groups and roles to any agent and maintains the correct information about the agents in each group and their roles.

5.4.2 Agent lifecycle management

The micro kernel manages the starting and stopping of the agents, and maintains reference tables of all the launched agents, it assigns a global address to each agent consisting of the kernel address and the agent address in the local kernel (AgentAddress).

5.4.3 Local message passing

The passage of messages between the agents is taken care of by the micro kernel of Mad-Kit, which distributes the messages between the local agents. A message is sent by copying it into the buffer of the receiving agent.

5.4.4 Systematic agentification of services

The set of services offered by the platform is individually embodied by an agent or a set of agents to provide services like monitoring the state of the system, connection to a network of kernels, agent migration, management of non-local messages, security of agent organizations, simulation tools and in general all applications and extensions of the platform. The *GroupeObserver* agent shown in the Figure 7 belongs to the MadKit platform. It provides the services to monitor the states of our simulator. In addition, it allows us to observe the membership of the roles of the agents in each group of our system, as well as the information on the various message exchanges between the agents of the system (sender, receiver, type of message, content, date and time of sending, etc.).

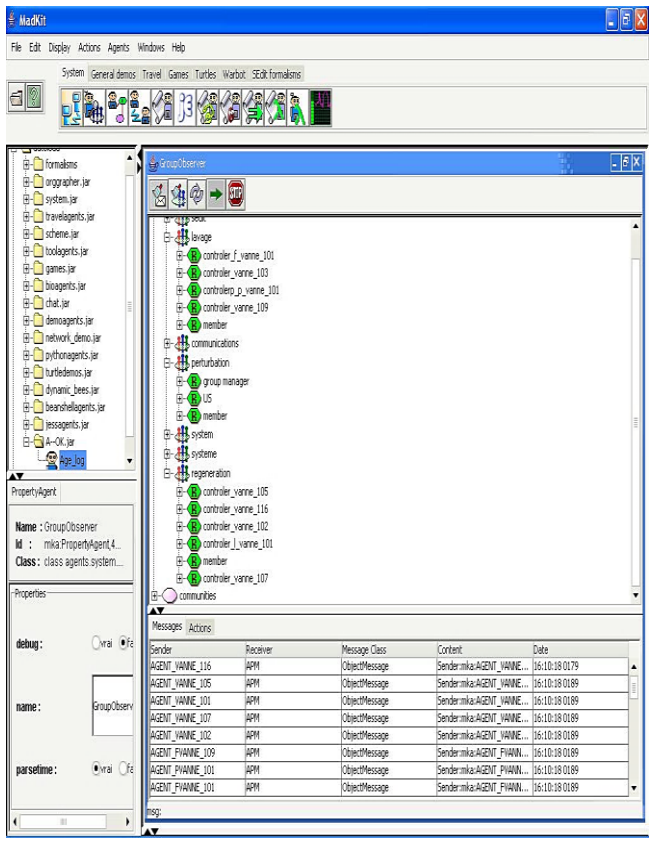

Figure 7. The group observer agents

5.5 The MADKIT agents

The important entity for programming under Mad-Kit is the AbstractAgent class. It defines the basic methods that an agent has: like the void walk () method for communication, but this class does not define its behavior. This task is devoted to the developer who defines the behavior of the agent according to his needs, that is to say the static and/or dynamic polymorphism of these methods. The agent's heirs to this class must be in permanent communication (sending and receiving messages). For this, they also necessarily inherit the ReferenceableAgent interface through which the micro kernel references the agents so that they are identifiable.

Example:

```
public class Agent_valve_fv_101 extends
AbstractAgent implements ReferenceableAgent
{                    }
```

- Activation section: which contains the code to be executed when the agent is created, this code is written in the activate() method.

- End section: It is represented by the voidend() method, it contains the code to be executed if the agent has finished processing or when the agent is killed.

Between the two previous sections the agent has a job to do and a mission to accomplish within the system. The behavior of an agent must be coded in the predefined void live() method for the heirs of the Agent class and in the void walk() method for heirs of the AbstractAgent class. These methods will be executed directly after the activation section, that is, during launch.

5.6 Managing groups and roles

The group creation is done by the createGroup (…) method. By example: createGroup (Boolean distributed, String comunauty1, String group1, String s, GroupIdentifier a).

This method is used to create the group group1 in the community community1 if this parameter is present otherwise the group is "public". The group is distributed if the Boolean parameter distributed is true and local (i.e. agents are not visible to remote kernels) otherwise. The parameter s is used to describe the overall structure of the group, but this device is not yet used. The parameter must be a GroupIdentifier, that is, an object that implements a security mechanism.

5.7 Message passing

Communication: The agent's interactions are made by exchange of messages. This exchange of information is done through the Message class. This class defines things like transmitter, receiver or the date of issuance.

Message passing: In sending of messages the sendMessage method (AgentAddress, message) is the most basic and lowest level. It sends the message to the agent at " AgentAdress" which may be acquired by way of any other agent via requesting both group and role. Two parameters are used:

- The first one is AgentAddress type that allows a unique identity of the address of the receiving agent. It is getting back thanks to the getAgentWithrole (group, role) method.

- The second suggests the message to be send. For example, the ObjectMessage class allows sending any object.

For the receiving messages, there are two methods:

-IsMessage Box Empty(): Checks if the agent has not read his mailbox messages, which returns true if it is empty and if there are unread messages it returns false.

-NextMessage() method for recovering the first unread, and to remove it from the message box.

Example:

```
ObjectMessage   obj = new
ObjectMessage((Object)message);
AgentAddress    adr =
getAgentWithRole(groupe,role);
sendMessage(adr,obj);
}
```

To read a message, the agent consults his mailbox via the isMessageBoxEmpty() method which returns true if it is empty, if there are unread messages it returns false, in this case the nextMessage() instruction removes and return the oldest unread message in the agent's message box.

Example: The following portion of the code shows the use of these methods by the APM agent

```
      public void walk()
      {    while (!isMessageBoxEmpty() )
             {                          message=
(ObjectMessage)nextMessage();

analyser_message(message);}}
```

5.8 The scheduling

The Scheduling under Mad-Kit is delegated to agents inheriting from the Scheduler class. The purpose of an agent scheduler is to coordinate the execution via generic tool objects called Activators, these are the means for the scheduler to identify a set of agents. The code of an Activator contains all the methods of the other agents that it will invoke when it is activated by the scheduler agent. Note here that the methods invoked by activators are those of agents which inherit from the AbsractAgent class and implement the ReferenceableAgent interface, and not the agents inheriting from the Agent class. Indeed, the use of schedulers and activators reduces the early CPU usage by the system agents, on the other hand, the use of schedulers and activators gives more flexibility. We have enabled an Agent Scheduler which uses a set of activators as a means to run the agents in our system.

The scheduler is composed of the following activators:

• An activator that executes the code of the disturbance agent.

• Two activators, one for processing and one for the graphical display, that run the APM agent.

• Two activators, one for treatment and one for the graphical display, which run the Agents in the Regeneration group.

• Two activators, one for processing and the other for graphical display, to run the Agents in the Washing group.

5.9 The simulator interaction interfaces

The two main interfaces of our project are presented in Figures 8 and 9. These two interfaces concern the washing section and the interface of the regeneration section. They allow the user to drive the system operation and supply it by loading different data inputs.

The simulator's graphical interfaces offer several tools allowing the user to control and monitor changes in the process. Among these parameters, we present a few below:

a) The input parameters:

- The operator can act on the input elements of the system (valves,…) by entering set point values and observe the behavior of the simulator accordingly.

- The simulator clock can be clocked in step-by-step mode: The simulator displays a state then waits for the operator to validate the clock to go to the next state. This allows the operator to understand the behavior of the simulator at each moment. Just as the operator can opt for an automatic

operation of the simulator and in this case the various states are displayed continuously at a rate defined by the operator.

The operator has the possibility to stop the progress of the simulator or to switch to step-by-step mode.

b) The displayed data:

Perhaps the most important part of the simulator is the display of information. Thanks to this interface, the operator can observe the steps of the process. Thus, several types of information have been provided for this purpose: - display of all the elements entering into the processing of the process; this is the display of the simulator synoptic.

- Visualization of the values of the processing elements in each state.

- Display of the current state of progress in the processing of the process.

c) Discussion:

As it is designed, the simulator offers a certain number of functionalities allowing the operator to follow the evolution of the process and to be able, thus, to understand its operation. Likewise, such a simulator makes it possible to predict several cases: those, which are acceptable, and those, which are prohibited. Additional features can be added depending on user needs.

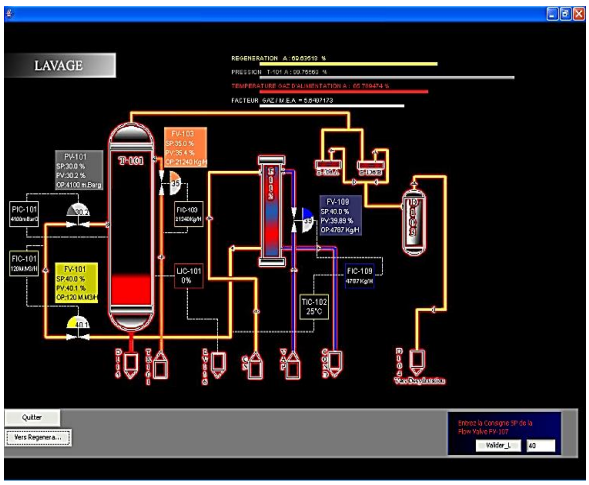

Figure 8. The interface of the washing section simulator

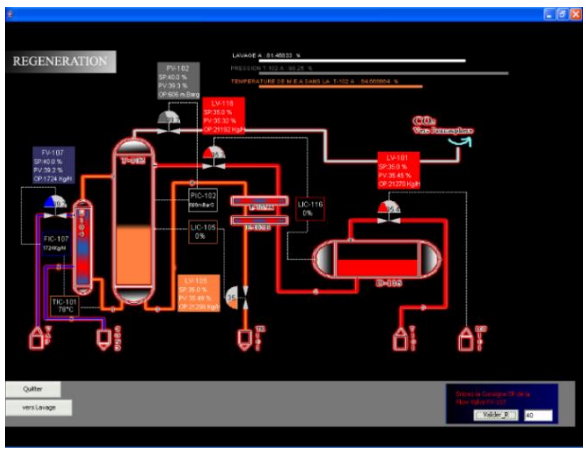

Figure 9. The interface of the regeneration section simulator

6. CONCLUSION

The work presented in this article concerned a top down approach to decompose a global complicated process into a set of its constituents. This demarche was adopted to distribute the complexity of the whole system. The obtained elementary elements are of fewer complexities. We choose the MAS based modelling and simulation approach, which is very suitable for this kind of modeling. Among the existing design, the development and the implementation tools; our choice was oriented towards the organizational model AALAADIN for the design and the MADKIT platform for the development and the implementation of our simulation model.

The concerns of this work is to introduce some notions of the design of industrial simulators. This work can be considered as an assistant of training and learning for beginner operators, workers new recruits and trainees enabling them to understand and simulate industrial processes without acting on the real system and without stopping processes and avoiding losses of time and money.

For its concerns, the realization of a concrete simulator intended for the treatment of a petrochemical process, in this case the decarbonation of liquefied natural gas has been implemented and is operational. The advantage of this implementation lies in the possibility of offering new recruits technological tools allowing them to have a precise idea of the different stages of the treatment of this process. Therefore, we can consider our simulator more as a teaching aid. The Multi-Agents approach is a high-level method for systems decomposition.

For future work, we plan to generalize our approach to encompass a set of applications in the field of natural gas processing. This work responds to increased demand from the industrial sector.

REFERENCES

[1] Abar, S., Theodoropoulos, G.K., Lemarinier, P., O'Hare, G.M. (2017). Agent based modelling and simulation tools: A review of the state-of-art software. Computer Science Review, 24: 13-33. https://doi.org/10.1016/j.cosrev.2017.03.001

[2] Bandini, S., Manzoni, S., Vizzari, G. (2009). Agent based modeling and simulation: an informatics perspective. Journal of Artificial Societies and Social Simulation, 12(4): 4.

[3] Seddari, N., Redjimi, M., Boukelkoul, S. (2014). Using of DEVS and MAS tools for modeling and simulation of an industrial steam generator. Journal of Computing and Information Technology, 22(3): 171-189. https://doi.org/10.2498/cit.1002348

[4] Seddari, N., Boukelkoul, S., Bouras, A., Belaoued, M., Redjimi, M. (2021). A new transformation approach for complex systems modelling and simulation: Application to industrial control system. International Journal of Simulation and Process Modelling, 16(1): 34-48. https://doi.org/10.1504/IJSPM.2021.113073

[5] Seddari, N., Redjimi, M. (2013). Multi-agent modeling of a complex system. In 2013 3rd International Conference on Information Technology and e-Services (ICITeS), pp. 1-6. https://doi.org/10.1109/ICITeS.2013.6624072

[6] Fishwick, P.A. (1995). Simulation Model Design and Execution: Building Digital Worlds. Prentice Hall PTR.

[7] Fishwick, P.A. (1997). Computer simulation: Growth through extension. Transactions of the Society for

Computer Simulation, 14(1): 13-24.
[8] Zeigler, B.P., Muzy, A., Kofman, E. (2019). Introduction to iterative system specification. Theory of Modeling and Simulation, 255-280. https://doi.org/10.1016/b978-0-12-813370-5.00019-5
[9] Mubarak, M., Carothers, C.D., Ross, R.B., Carns, P. (2016). Enabling parallel simulation of large-scale HPC network systems. IEEE Transactions on Parallel and Distributed Systems, 28(1): 87-100. https://doi.org/10.1109/TPDS.2016.2543725
[10] Wainer, G., Liu, Q., Jafer, S. (2018). Parallel simulation of DEVS and Cell-DEVS models in PCD++. Discrete-Event Modeling and Simulation, 223-270.
[11] Martınez-del-Amor, M.A., Riscos-Núnez, A., Pérez-Jiménez, M.J. (2017). A survey of parallel simulation of P systems with GPUs. Bull. Int. Membr. Comput. Soc. (IMCS), 3: 55-67.
[12] Juranic, J., Pavkovic, N., Naumann, T., Marjanovic, D. (2017). Modelling the design parameters dynamics with Petri nets. In DS 87-2 Proceedings of the 21st International Conference on Engineering Design (ICED 17) Vol 2: Design Processes, Design Organisation and Management, Vancouver, Canada, pp. 091-100.
[13] Jensen, K. (1987). Coloured petri nets. In Petri nets: Central Models and Their Properties, pp. 248-299. https://doi.org/10.1007/BFb0046842
[14] Jensen, K., Kristensen, L.M., Wells, L. (2007). Coloured Petri Nets and CPN Tools for modelling and validation of concurrent systems. International Journal on Software Tools for Technology Transfer, 9(3): 213-254. https://doi.org/10.1007/s10009-007-0038-x
[15] Sunitha, E.V., Samuel, P. (2019). Automatic code generation from UML state chart diagrams. IEEE Access, 7: 8591-8608. https://doi.org/10.1109/ACCESS.2018.2890791
[16] Mess, M., Guerrits, B. (2018). Multi-agent Systems. In: Zijm H., Klumpp M., Regattieri A., Heragu S. (eds) Operations, Logistics and Supply Chain Management. Lecture Notes in Logistics. Springer, Cham. https://doi.org/10.1007/978-3-319-92447-2_27
[17] Ferber, J. (1995). Les Systèmes Multi-Agents: Vers Une Intelligence Collective. Informatique, Intelligence Artificielle, Intereditions Paris.
[18] Ferber, J., Gutknecht, O., Michel, F. (2003). From agents to organizations: an organizational view of multi-agent systems. In International Workshop on Agent-Oriented Software Engineering, pp. 214-230. https://doi.org/10.1007/978-3-540-24620-6_15
[19] Souidi, M.E.H., Songhao, P., Guo, L., Lin, C. (2016). Multi-agent cooperation pursuit based on an extension of AALAADIN organisational model. Journal of Experimental & Theoretical Artificial Intelligence, 28(6): 1075-1088. https://doi.org/10.1080/0952813X.2015.1056241
[20] Russell, S.J. (1997. Rationality and intelligence. Artificial Intelligence, 94: 57-77.
[21] Ferber, J., Gutknecht, O. (1998). Aaladin: A meta-model fort the analysis and design of organizations in multi-agent systems. In Les Actes de 3rd International Conference on Multi-Agent Systems (ICSMAS'98), pp. 128-135.
[22] Chebout, M.S., Mokhati, F., Badri, M., Babahenini, M.C. (2019). Monitoring open multi-agent systems: An aspect-oriented programming based approach. Multiagent and Grid Systems, 15(2): 155-177. https://doi.org/10.3233/MGS-190307
[23] http://mansour.saber.free.fr/termadkit/site/madkit/doc/devguide/devguide.html, accessed on September 24 2020.
[24] Abar, S., Theodoropoulos, G.K., Lemarinier, P., O'Hare, G.M. (2017). Agent based modelling and simulation tools: A review of the state-of-art software. Computer Science Review, 24: 13-33. https://doi.org/10.1016/j.cosrev.2017.03.001
[25] Bandini, S., Manzoni, S., Vizzari, G. (2020). Agent-based modeling and simulation. Complex Social and Behavioral Systems: Game Theory and Agent-Based Models, pp. 667-682.
[26] Chen, S., Zhang, H., Guan, J., Rao, Z. (2020). Agent-based modeling and simulation of stochastic heat pump usage behavior in residential communities. In Building Simulation, 13(4): 803-821. https://doi.org/10.1007/s12273-020-0625-2
[27] Mahmood, I., Mobeen, M., Rahman, A.U., Younis, S., Malik, A.W., Fraz, M.M., Ullah, K. (2020). Modeling, simulation and forecasting of wind power plants using agent-based approach. Journal of Cleaner Production, 276: 124172. https://doi.org/10.1016/j.jclepro.2020.124172
[28] Macal, C.M. (2020). Agent-based modeling and artificial life. Complex Social and Behavioral Systems: Game Theory and Agent-Based Models, 725-745. https://doi.org/10.1007/978-1-0716-0368-0_7
[29] Herrera, M., Pérez-Hernández, M., Kumar Parlikad, A., Izquierdo, J. (2020). Multi-agent systems and complex networks: Review and applications in systems engineering. Processes, 8(3): 312. https://doi.org/10.3390/pr8030312
[30] Zhou, Z., Chan, W.K.V., Chow, J.H. (2007). Agent-based simulation of electricity markets: A survey of tools. Artificial Intelligence Review, 28(4): 305-342. https://doi.org/10.1007/s10462-009-9105-x
[31] Yang, Y. (2020). Research on modeling and simulation of agent-based intelligent teaching system. Science Journal of Education, 8(1): 27-31. https://doi.org/10.11648/j.sjedu.20200801.15
[32] Savaglio, C., Ganzha, M., Paprzycki, M., Bădică, C., Ivanović, M., Fortino, G. (2020). Agent-based Internet of Things: State-of-the-art and research challenges. Future Generation Computer Systems, 102: 1038-1053. https://doi.org/10.1016/j.future.2019.09.016
[33] Fellir, F., El Attar, A., Nafil, K., Chung, L. (2020). A multi-Agent based model for task scheduling in cloud-fog computing platform. In 2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT), pp. 377-382. https://doi.org/10.1109/ICIoT48696.2020.9089625
[34] Nanna, G.A., Quatraro, N.F., De Carolis, B. (2020). A multi-agent system for simulating the spread of a contagious disease. Session 5: Agents & Actors for Data Science 160, 1613: 119.
[35] Chen, F., Ren, W. (2019). On the control of multi-agent systems: A survey. Foundations and Trends® in Systems and Control, 6(4): 339-499. http://dx.doi.org/10.1561/2600000019
[36] Weiss, G. (1999). Multiagent systems: a modern approach to distributed artificial intelligence. MIT Press.
[37] Wang, B., Wang, J., Zhang, B., Li, X. (2016). Global cooperative control framework for multiagent systems

subject to actuator saturation with industrial applications. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 47(7): 1270-1283. https://doi.org/10.1109/TSMC.2016.2573584
[38] Ferber, J. (1999). Multi-Agent Systems. An Introduction to Distributed Artificial Intelligence. Addison-Wesley.
[39] Shoham, Y. (1993). Agent-oriented programming. Artificial Intelligence, 60(1): 51-92. https://doi.org/10.1016/0004-3702(93)90034-9
[40] Franklin, S., Graesser, A. (1996). Is it an Agent, or just a Program? A Taxonomy for Autonomous Agents. In International Workshop on Agent Theories, Architectures, and Languages, pp. 21-35. https://doi.org/10.1007/BFb0013570
[41] Wooldridge, M., Jennings. N.R. (1995). Agent theories, architectures, and languages. In Wooldridge and Jennings, eds. Intelligent Agents, Springer Verlag, pp. 1-22.
[42] Russell, S., Norvig, P. (2010). Artificial Intelligence: A Modern Approach, Pearson.
[43] Ye, P., Wang, S., Wang, F.Y. (2017). A general cognitive architecture for agent-based modeling in artificial societies. IEEE Transactions on Computational Social Systems, 5(1): 176-185. https://doi.org/10.1109/TCSS.2017.2777602
[44] Gutknecht, O., Ferber, J., Michel, F. (2000). MadKit: une architecture de plate-forme multi-agent générique (Doctoral dissertation, Lirmm, University of Montpellier).
[45] Davis, R. (1980). Report on the Workshop on Distributed AI. https://dspace.mit.edu/bitstream/handle/1721.1/41155/AI_WP_204.pdf?sequence=4.
[46] Konolige, K., Nilsson, N.J. (1980). Multiple-agent planning systems. Proc. AAAI 1980, 80: 138-142.
[47] http://www.madkit.net/madkit/, accessed on September 24, 2020.
[48] https://patents.google.com/patent/WO2014199036A2/en, accessed on September 24 2020.