World Academy of Science, Engineering and Technology
International Journal of Industrial and Manufacturing Engineering
Vol:11, No:10, 2017

Agent/Group/Role Organizational Model to Simulate an Industrial Control System

Noureddine Seddari, Mohamed Belaoued, Salah Bougueroua

***Abstract*—**The modeling of complex systems is generally based on the decomposition of their components into sub-systems easier to handle. This division has to be made in a methodical way. In this paper, we introduce an industrial control system modeling and simulation based on the Multi-Agent System (MAS) methodology AALAADIN and more particularly the underlying conceptual model Agent/Group/Role (AGR). Indeed, in this division using AGR model, the overall system is decomposed into sub-systems in order to improve the understanding of regulation and control systems, and to simplify the implementation of the obtained agents and their groups, which are implemented using the Multi-Agents Development KIT (MAD-KIT) platform. This approach appears to us to be the most appropriate for modeling of this type of systems because, due to the use of MAS, it is possible to model real systems in which very complex behaviors emerge from relatively simple and local interactions between many different individuals, therefore a MAS is well adapted to describe a system from the standpoint of the activity of its components, that is to say when the behavior of the individuals is complex (difficult to describe with equations). The main aim of this approach is the take advantage of the performance, the scalability and the robustness that are intuitively provided by MAS.

***Keywords*—**Complex systems, modeling and simulation, industrial control system, MAS, AALAADIN, AGR, MAD-KIT.

I. Introduction

COMPLEX systems are made from many elementary components in interaction, where the overall behavior emerges from the interaction of the entities which make it up. Both modeling and simulation of this type of systems [1]-[10] make it possible to handle, observe, and improve understanding the phenomena. Those items are studied in different levels as for the simulation of weather changes, the physical phenomena, and systems of urban traffic. Most of the modeled phenomena reach today complexities and high degrees of smoothness which requires the use of models, and thus, the data-processing tools become more and more flexible and performing.

The simulation consists of developing the model for various entries and observing exit points at precise moments, and durations to get a set of conclusions. According to Shannon‘s view [3], simulation is defined as follows: ' the process of designing a model of a real system and conducting experiments with this model for purpose either of understanding the behavior of the system of evaluating various strategies (within the limits imposed by a criterion or a set of criteria) for the operation of the system'.

According to Fishwick [7], the data-processing simulation can be understood 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'.

Our work may be defined in these terms: Which the steps are to be followed and the tools to be used for modeling and simulation of an industrial process?

Today, there exist a large and significant panoply of methods, tools, products, platforms and development environments intended for process’ modeling and simulation. Thus, the work that we are presenting in this paper is dealing with the modeling of the various elements being implemented in an industrial process. It concerns passive items (electro-valves, pumps, turbo-compressors, boilers, ventilators...), active items (computers, regulators, controllers, servers...) and elements for communication (pipes, wire, control and command signals, alarms, display screen and synoptic, keyboards...) along with elements of memorization, storage, warehousing and data processing.

According to its importance, an industrial system can contain thousands of elements and even more. This is the case, for example, of the complexes of petroleum processing.

The present work in this paper comes in this context and we are based on the organizational model AALAADIN [11]; this model is a project that focuses on the analysis, design, formalization and implementation of MAS in organizational prospects. The underlying conceptual model (AGR) is based on the concepts of agents, groups and roles [15].

This model was developed by the team of Professor J. Ferber with the [11], [12]. Within the scope of this implementation, our model will be implemented on the MAD-KIT multi-agents platform for its simplicity of implementation and the tools which make it available.

This paper is organized as follows: Section II will present the problematic behind our contribution. Section III is a brief presentation of MAS. Section IV illustrates an overall description of the industrial system to be simulated. The Multi-Agents model of this process is described in Section V. The implementation of the system is shown in Section VI. Finally, we conclude this paper in Section VII.

II. Problem and Motivation

Modern companies are based on very complex industrial

Noureddine Seddari was with University of Skikda, Skikda, Algeria. He is now with the Department of Fundamental Computer Science and its Applications, Constantine 2 University, Constantine, Algeria (e-mail: noureddine.seddari@univ-constantine2.dz).

Mohamed Belaoued is with LIRE Laboratory, Constantine 1 University, Constantine, Algeria (e-mail: belaoued.mohamed@gmail.com).

Salah Bougueroua is with the Department of Computer Science, University of Skikda, Skikda, Algeria (e-mail: s_bougueroua@yahoo.fr).

systems. One of the main reasons of the complexity of these industrial systems is that they are often themselves systems of systems, and they are therefore the result of the integration of many other industrial systems relatively complex. Thus, an industrial system is often classified as a complex system when the hierarchical decomposition of such system reveals a large number of sub-systems of many different natures. The main sources of complexity of an industrial system are its size, heterogeneity and the large number of interactions between its components. These parameters are also most often present simultaneously in a complex industrial system.

The modeling of such systems requires a rigorous approach consisting in:

- Determination and decomposition of the parts of the system to be modeled and simulated.
- Determination of the atomic and coupled models which compose the system.
- For each model, we need to precise the sequence of states, connections, communications, interactions and interfaces with others related models.
- The atomic and coupled models which compose the system to be simulated are represented by agents and groups of agents respectively [17].

In our adopted method, we have used the MAD-KIT platform, particularly for the concepts of Agent, Group and Role (AGR). This platform offers highly successful tools for the management of the agents (creation, scheduling, interactions with messages passing...).

III. Modeling by Using MAS

The topic of MAS is at the root of the connection of several specific fields of artificial intelligence, distributed computing systems and software engineering. It is a discipline that focuses on collective behavior produced by interactions of several autonomous and flexible entities called agents. Those interactions turn around the cooperation, competition or coexistence between those agents.

A MAS is a distributed system consisting of a set of agents. MAS are designed and implemented as a set of interacting agents, most often in ways of cooperation, competition or coexistence.

MAS is basically characterized by [13]:

Each agent has limited information and problem-solving abilities, and each agent has an individual space (its environment) and a partial view of a distributed space (sharing environment);

- ✓ There is no overall control of MAS.
- ✓ Data are dislocated.
- ✓ Calculations are asynchronous.

There are several ways of representing and formalizing an agent, thus Ferber represents a MAS by the couple < A, W > where A is an agent and W an environment [14]–[16].

$$A = (Pa,\ Percepta,\ Fa,\ Infla,\ Sa) \quad (1)$$

with P_a: function of perception of the agent. Percept $_a$: A set of stimuli and feelings which an agent can perceive. F_a: function of behavior of the agent. $Infl_a$: function of action of the agent. S_a: A set of internal states of agent.

$$W = (E, \Gamma, \Sigma, R) \quad (2)$$

with E: space in which the agent evolves. Γ: space of influences produced by the agent. S: state of the environment. R: law of evolution.

Although the MAS offer many potential advantages, they must also raise many challenges. Here are some defies in the MAS implementation [18]-[20]:

- How to formulate, describe, decompose, and allocate the problems and to synthesize the results?
- How to allow the agents to communicate and interact? What and when to communicate?
- How to ensure that the agents act in a coherent way i) taking their decisions or actions, ii) managing the nonlocal effects of their local decisions and iii) avoiding the harmful interactions?
- How to allow the individual agents to represent and reason on the actions, plans and knowledge of other agents in order to coordinate with them? How to reason on the state of their coordinated processes (such as initialization or termination)?
- How to recognize and reconcile the disparate view points and the intentions conflict in a set of agents trying to coordinate their actions?
- How to find the best compromise between the local processing at one agent and distributed processing between several agents (distributed processing which induces communication)? More generally, how to manage the allocation of resources limited?

A. AALAADIN Model

The organizational model AALAADIN is a project that focuses on the analysis, design, formalization and implementation of MAS in organizational prospects.

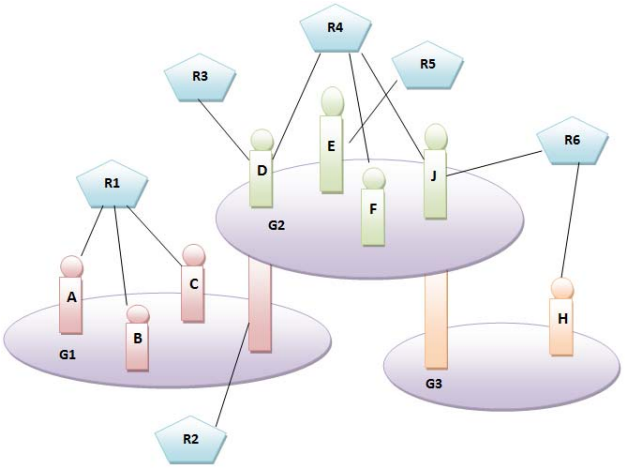

Fig. 1 The "cheeseboard" notation

Several notations may be used to represent organizations. The following representation, that we call the cheeseboard

diagram. This is a very convenient notation to represent examples of concrete organizations [21].

In this notation, A group is represented as an oval that looks like a board: A group is a set of atomic agents. The concept of the group allows labeling a set of agents. Every agent has the ability to create a group.

Agents are represented as skittles: An agent is only specified as a communicating entity which plays one or several roles in one or several groups. This definition, broadly speaking, is voluntary and this leads freedom to designers.

A role is represented as a hexagon and a line links this hexagon to agents: The role is the abstraction of agent's operation (either the service is providing or its identification) within a group. An agent may have several roles, and each role is local to a group. An agent may request a role in a group, but it will not be necessary assigned. Fig. 1 gives an example of a concrete organization using the cheeseboard diagram. In this picture, the agent D is a member of both G1 and G2, playing roles R3 and R4 in G2, and R2 in G1.

IV. Description of the System to Be Simulated

Our work is realized in close collaboration with SONATRACH complex and in particular the Liquefied Natural gas (LNG) units located in Skikda in the east of Algeria. The aim was to realize operational simulators of distributed Control Systems (DCS) of industrial processes used in this complex [22], [23].

Three major components of process to be modeled and simulated are as follows: the compressor, the boiler, and the water supply.

A. The Compressor

Its role is to ensure the operation of compression of the steam coming from the boiler. Fig. 2 represents the technical drawing of this compressor.

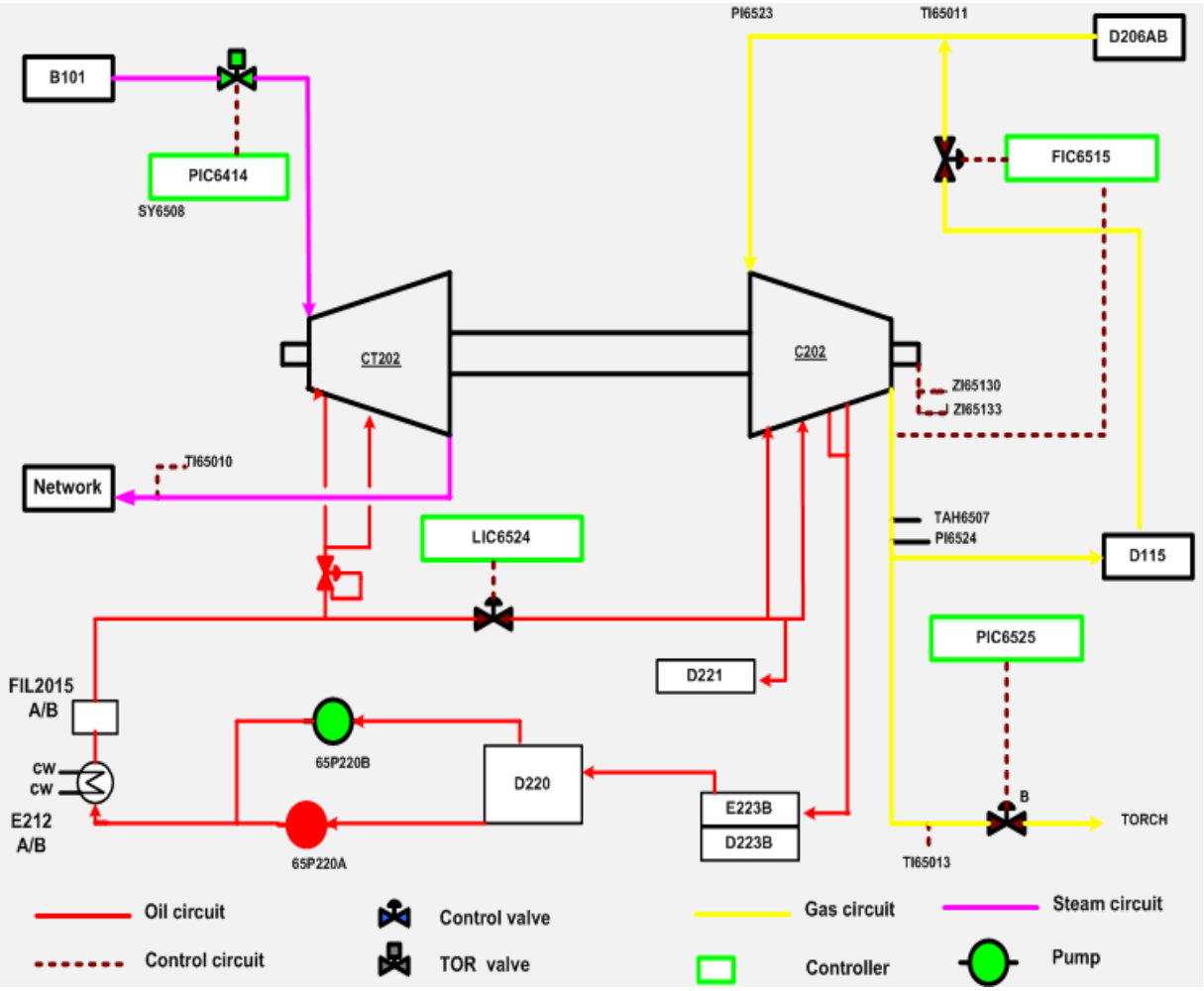

Fig. 2 Compressor section

Regulations ensuring the compression of steam: The pressure of steam is regulated by a loop of regulation PIC6414.

The gas flow entering towards the compressor is regulated by a loop of regulation LIC6515.

The oil flow is regulated by a loop of regulation LIC6524.

B. The Boiler

A boiler is a steam generator; it aims to raise the temperature of water until the change of its status, that is to say, to become steam and then to bring it to the properly defined pressure and temperature. Fig. 3 shows the technical drawing of this process.

As boiler construction, we distinguish:

- A metal structure with its trim, masonry casing.
- The boiler: tubes, box back, screens, spray, super heaters, balloons…
- The combustion chamber: burners, air and auxiliary circuits.

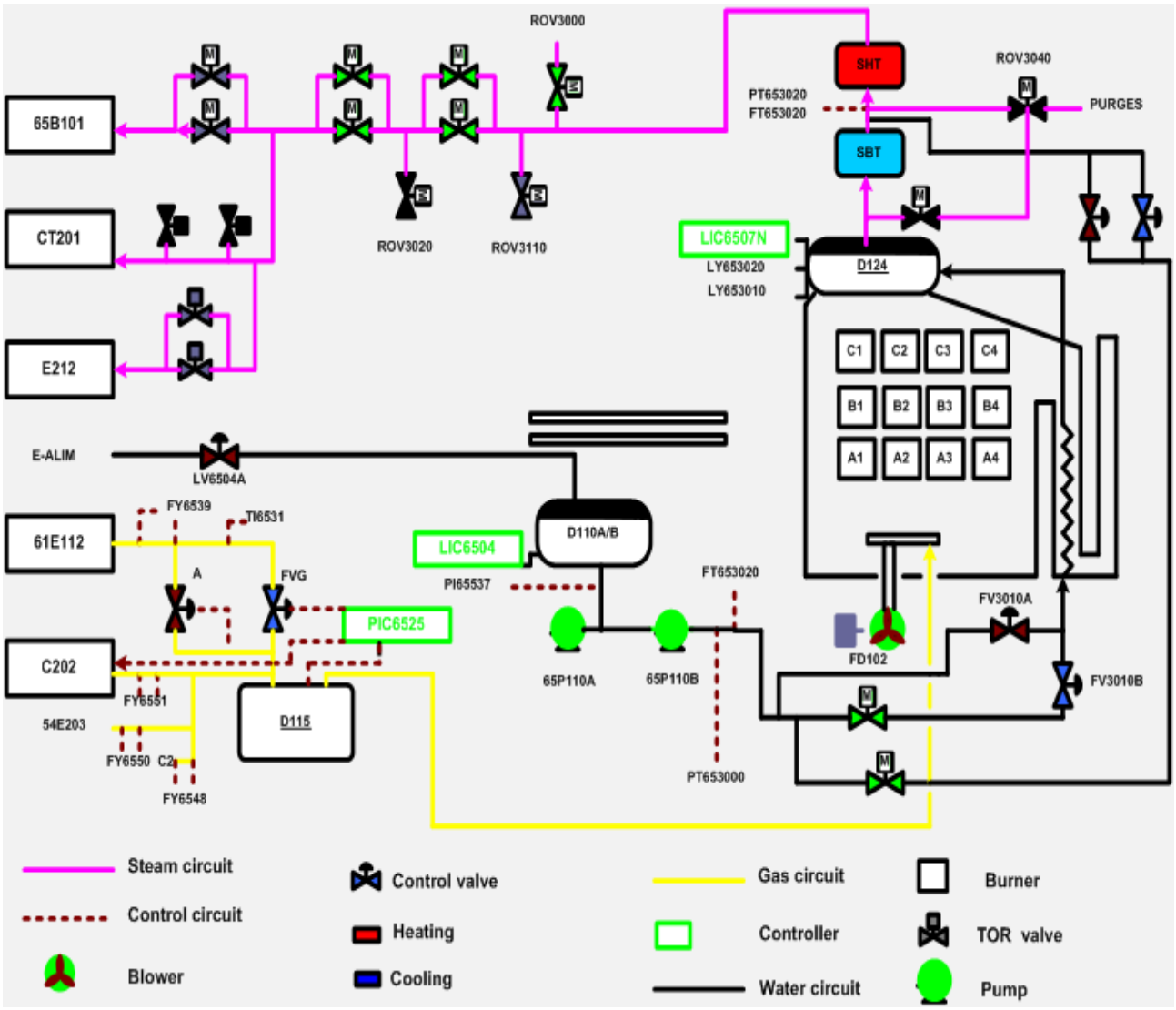

Fig. 3 Boiler section

Regulations which ensure the generation of steam: The combustion of gas is regulated by a loop of regulation PIC6525.

The level of the balloon is regulated by a loop of regulation LIC6507N.

C. The Water Supply

The water supply is a system that feeds the boiler. Fig. 4 shows the technical drawing of this section.

Regulations ensuring the feeding of water: The level of water in the condenser (E212) is regulated to 65% by a loop of regulation LIC6501.

The level of water in the feeding tank (D110A/B) is regulated to 65% by a loop of regulation LIC6504.

V. Proposal for AGR Model of the Simulator

To model our system, we have used the organizational model AALAADIN (Ferber and Gutknecht 1998), which establishes the basis of our design. Therefore, we have split up our process into three groups each group contains a set of the agents. In the proposed decomposition, we have modeled just the sensitive components such as gas electro-valve (FVG, FIC6515), combustion regulator (PIC6525), air ventilator (FD112), water electro-valves (LV6504A, FV3010B, 15FVA,…), water levels regulators (LIC6504, LIC6501,

LIC6507N…)…

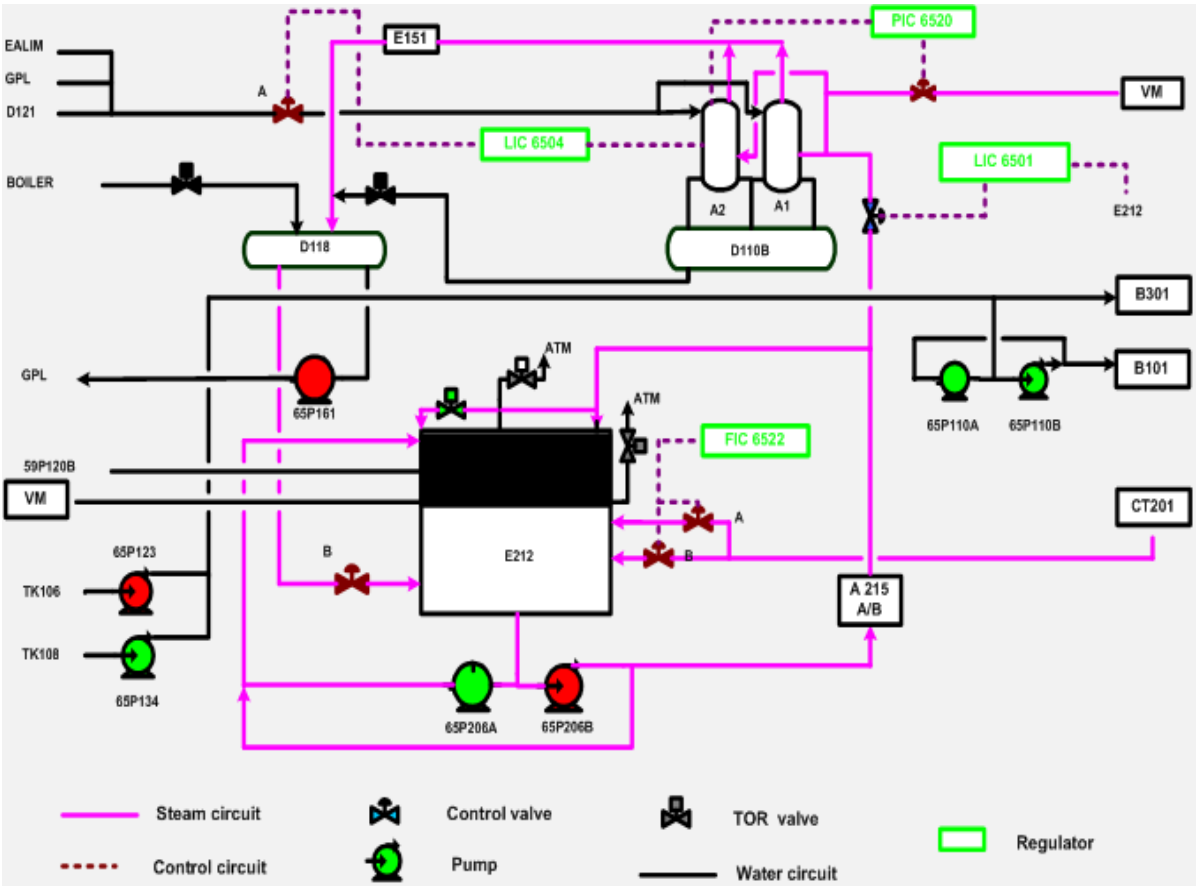

Fig. 4 Water supply section

A. Groups and Roles of the System

The compressor group: This group contains a set of agents in interaction making possible to execute the operation of steam compression.

- *Agent (PIC6414):* Is in charge of the steam pressure. Its role is to calculate this pressure by using the necessary values, provided by the other agents of the system.
- *Agent (FIC6515):* Its role is to control the gas flow entering towards the compressor.
- *Agent(LIC6524):* The role of this agent is to control the oil level entering towards the compressor.

The boiler group: This group contains a set of agents in interaction making possible to execute the operation of steam generation.

- *Agent (PIC6525):* The role of this agent is to control the combustion and to regulate the measured sizes which are received constantly from various agents of system. Indeed, if these measurements are not consistent with the load of the boiler, it periodically sends orders of regulation towards the ventilator agent (FD112) and the gas valve agent that is to say agent (FVG) in order to reach desired sizes.
- *Agent (FD112):* Its role is to control the air flow; it sends periodically this flow towards the combustion agent (PIC6525) and receives the orders coming from this agent.
- *Agent (FVG):* This agent controls the gas flow; it sends periodically this flow towards the combustion agent (PIC6525) which transmits in its turn orders to it (opening, closing) in order to modify the gas output.
- *Agent (LIC6507N):* Its role is to regulate the level of the balloon by using values provided by the various agents of the system; the Hydraulic (LIC6507N) agent sends the orders of opening and closing towards the water valve agent that is to say agent (FV3010B) till this measured level becomes equal at the desired level (50%).
- *Agent (FV3010B):* The role of this agent is to control the flow of feed water tank coming from feed water station, it sends periodically this flow towards the agent balloon (D124) and the hydraulic agent (LIC6507N) by receiving orders (opening, closing) coming from the latter.
- *Agent (D124):* Its role is to calculate the level of balloon; it sends periodically this level towards the hydraulic agent (LIC6507N) to be controlled.

The water supply group: This group contains a set of agent in interaction by ensuring the operation of water alimentation of the boiler.

- *Agent (LIC6504*): Is in charge of the level of the tank, its role relates to the regulation of the tank by using the

necessary values provided by the other agents of the system, the feeding agent (LIC6504) sends the orders of opening and closing thanks to the agent Valve- tank (LV6504) in order to modify the flow of water and therefore modifies the level of the tank (15B01).

- *Agent (D110A/B):* It shows the variation of the level of water inside the tank, it communicates with the feeding agent by sending periodically the level in order to be regulated, and by the way it receives orders of the Feeding agent (LIC6504).
- *Agent (LV6504A):* This agent controls the flow of water entering the condenser (E212); it is in touch with the tank agent (D110A/B) and condensation agent (LIC6501) by providing water flow.
- *Agent (LIC6501): It* is in charge of condenser level (E212), its role relates to the regulation of the condenser by using the values provided by the other agents of the system, the Condensation agent (LIC6501) sends the orders of opening and closing by using the condenser valve agent (15FVA) in order to modify the water flow and to modify the level of the condenser.
- *Agent (E212): It* shows the variation of the level of water in the condenser, it communicates with the condensation agent (LIC6501) by sending periodically this level for which is regulated and it receives orders of the condensation agent.
- *Agent (15FVA): This* agent controls the water flow coming out the tank (D110A/B); it is in touch with the Condenser agent (E212), the Condensation agent (LIC6501) and the Feeding agent (LIC6504) by providing the water flow.

Fig. 5 represents the organizational structure of the system based on the suggested AGR model.

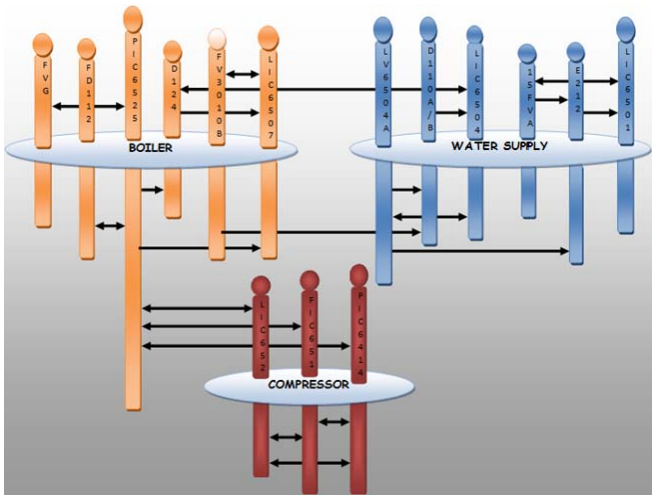

Fig. 5 AGR organizational structure of the simulator

VI. Implementation of the System

We will develop our "multi-agents simulator," using MAD-KIT platform to benefit from the power of the advanced tools offered by this platform for the implementation, the development and implementation.

This environment is based on three principles:

- Micro-kernel architecture
- Agentification of services
- Graphic component model

A. Agent Micro-Kernel

MAD-KIT micro-kernel is an environment for implementing the agents of reduced size, it provides the following functions [24], [25]:

Control of local groups and roles: The micro-core is in charge of maintaining accurate information about the group members and the roles handled. It checks also whether the requests made on the system of groups and roles are acceptable.

Agent life-cycle management: The micro-core launches the agents; it may be either hung or stopped and assigns them unique identifiers.

Local message passing: The kernel is in charge of distribution of messages between local agents (implemented on a kernel). The messages sent from one agent to another are given by the kernel if the receiver and transmitter are local, if otherwise; the messages will be delivered possibly by a specialized system agent.

Agents: The Abstract Agent class, it is the main class that supports the life cycle of agents, the groups management and roles as well as the messaging system, the graphical interface along with information about agents (address, the role ...). The portion of the following code (Fig. 6) shows the use of such classes by the D110A/B agent:

```
import madkit.kernel.*;
import madkit.lib.messages.StringMessage;
import madkit.lib.messages.ObjectMessage;


public class A_D110A/B extends AbstractAgent implements ReferenceableAgent
{
Graphics_Class syn_D110A/B;
 Graphics2D D110A/B_g_p_water;
 inter f_D110A/B;
 static int level_b1=65;
 int level_b2;
 int vap;
    int x;
    int i;.
    int j;

    int h;
    int l;
    int t_x;
    int t_y;


    ObjectMessage m_LIC6504;
    Grand t_D110A/B;
    int water_E212;
    int water_D110A/B;
    int water;

    calculate_level calculate_level;
    int variation;
*
*
*
```

Fig. 6 Parts of D110A/B java code

Control and life-cycle: Within MAS, it is far necessary to control a huge range of granularity agents and to manipulate their scheduling.

It is almost impossible if we are based on a process mechanism due to induced overload. In this case, we use "synchronous" agents via external agents that will define their synchronous execution policy. For implementation object of

execution policy, we set tiers, scheduler and activator.

The scheduling is delegated to the agents inheriting the Scheduler class. A Scheduler agent aims to coordinate the execution of agents through generic tools referred to as activators; the latter is the manner for the scheduler to identify a set of agents. The Java codes below (Figs. 7 and 8) show the use of schedulers and activators by agents of our simulator:

```
public class shed extends Scheduler
{
    agent_Simulateur ags;
    Activ_T_Com atc;
    Activ_T_Boi atb;
    Activ_T_Wat atw;
          int delay;
    //-------------------------------------------------------------------------\\
    public shed (agent_Simulateur ass)
    {
        ags=ass;
    }
//-------------------------------------------------------------------------\\
    public void live()
    {
        atc=new Activ_t_Com("compression","agent_traitement_compression");
        atb=new Activ_t_com("boiler","agent_traitement_boiler");
        atw=new Activ_t_com("water_station","agent_traitement_water_station");
        addActivator(atc);
        addActivator(atb);
        addActivator(atw);
        update();
            do{
            delay++;
            if (delay==101)delay=1;
            if(!agent_simulation.stop)
             if (delay%50==0)
              atc.execute();

           // agc.execute();

           pause(10);
          }
        while(true);
        }
    *
    *
```

Fig. 7 Parts of scheduler java code of simulator

```
import java.awt.*;
import javax.swing.*;
import madkit.kernel.*;
import java.awt.image.*;
import java.awt.event.*;
import madkit.lib.messages.ObjectMessage;
import madkit.lib.messages.StringMessage;
import java.util.*;


public class Activ_T_Com extends Activator
{

int cycle;
    public Activ_T_Com(String group, String role)
    {
       super(group, role);
    }


  public void execute()
  {
    cycle++;
    //if (cycle>=10)cycle=0;
    //if(cycle%2==0)
        agent_com.agent_PIC6414.t_msg();

        agent_com.agent_FIC6515.t_msg();

        agent_com.agent_LIC6524.t_msg();


  }
}
```

Fig. 8 Parts of activator java code of simulator

Communication: The aim of the agents is nevertheless to exchange basically messages. This exchange of information is done through the Message class. This class defined only 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 "message" to the agent at AgentAdress" which may be acquired by way of any other agent via requesting both group or the role. This approach uses two parameters:

- The first one is AgentAddress type that allows a completely unique identity of the address of the receiving agent; we get back thanks to the method: getAgentWithrole (group, role).
- The second suggests the message to be send; the ObjectMessage class for example allows sending any object which gives great freedom to the programmer to build the message.

In receiving messages, the agent has a mailbox wherein the position of the kernel messages, there are two methods of receiving messages:

- 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 of recovering the first unread, and removes it from the message box [17].

Example:

Fig. 9 shows the use of such methods by the FVG agent:

```
//-------------------------------------------------------------------------\\
public void t_msg()
 {
     while(!isMessageBoxEmpty())
   {
   m_FVG=(ObjectMessage)nextMessage();
   check_message(m_FVG);
                 };
        if(sec)
      if(op_FVG>10)
      op_FVG-=10;
      if(op_FVG<=10)op_FVG=0;
      calculate_flow();
      if(op_FVG==0)v_FVG=0;
      send_measurement();

      FVG=v_FVG;
 }
//-------------------------------------------------------------------------\\
void check_message(ObjectMessage m)
  {
    Grand t_FVG=(Grand)m.getContent();
    if(t_FVG.gettype()=="closing")
    {
       op_FVG-=var;
    }
       if(t_FVG.gettype()=="opening")
    {
       op_FVG+=var;
    }
    if(t_FVG.gettype()=="stop_boiler")
    {
    sec=true;
    }
      *
      *
```

Fig. 9 Parts of FVG Java code

We will present three main interfaces of our future simulator, the interface of the boiler, compressor and water supply. They allow the user to drive the regeneration steam operation, supply boiler with water and compression of steam by publishing guidance to various regulatory bodies and follow the operation of the process functioning as shown by the following figures:

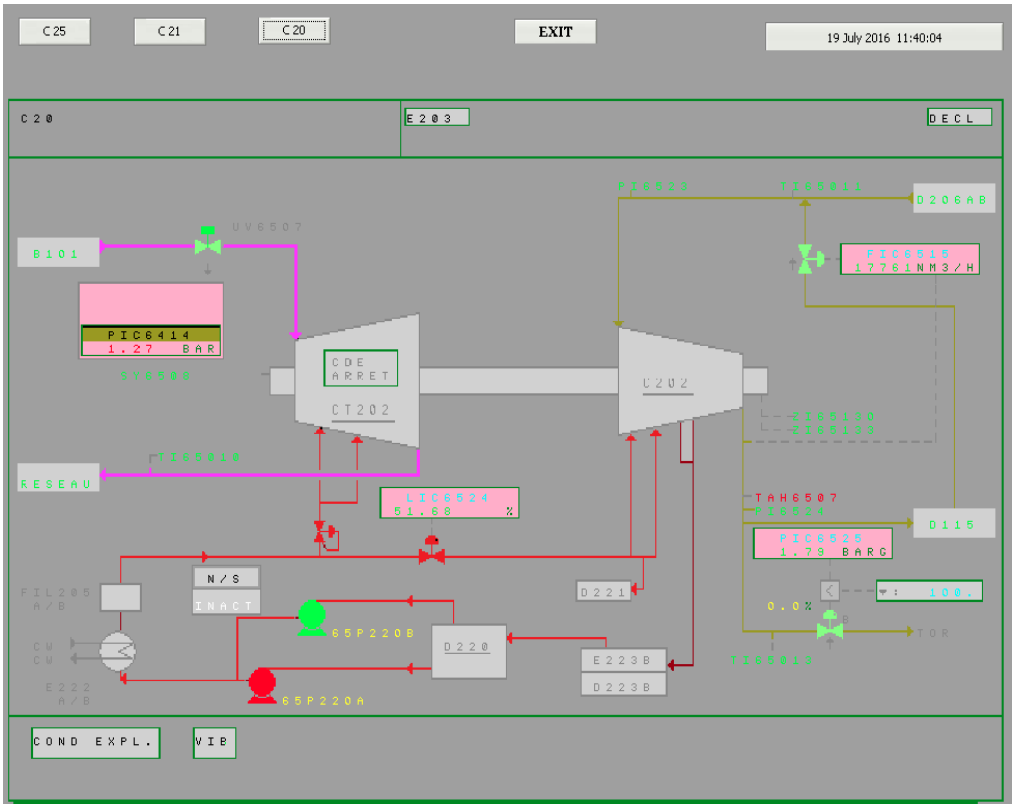

Fig. 10 Graphical simulation of the compressor

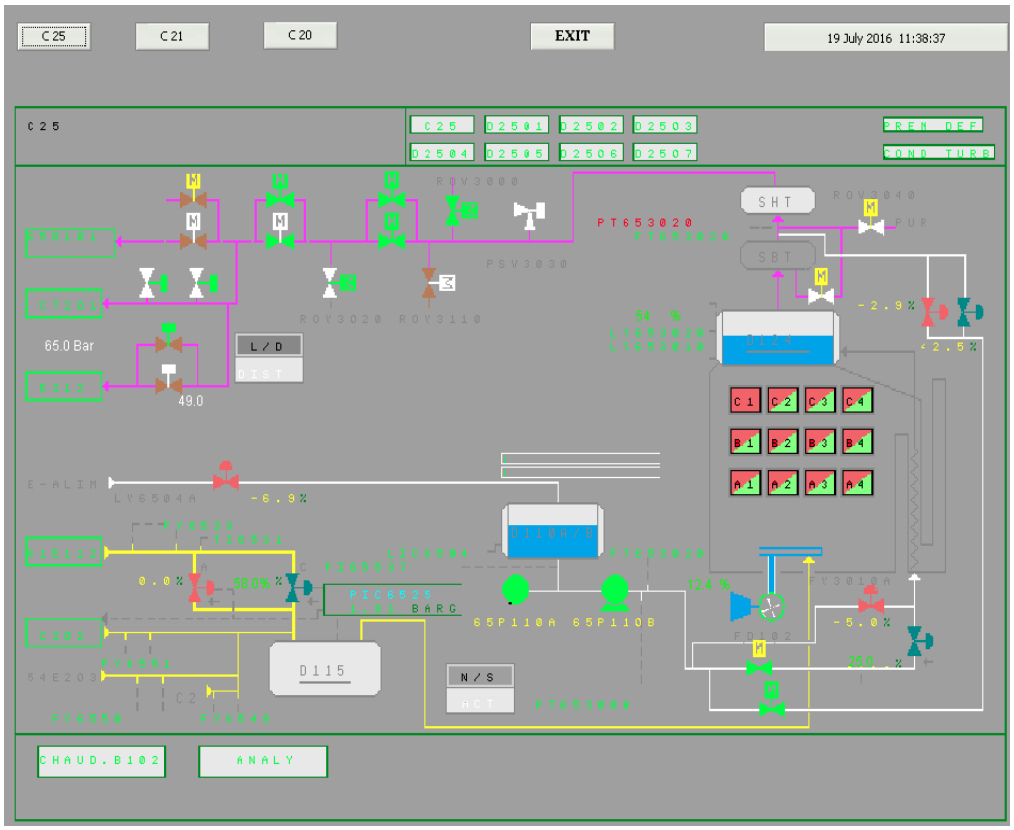

Fig. 11 Graphical simulation of the boiler

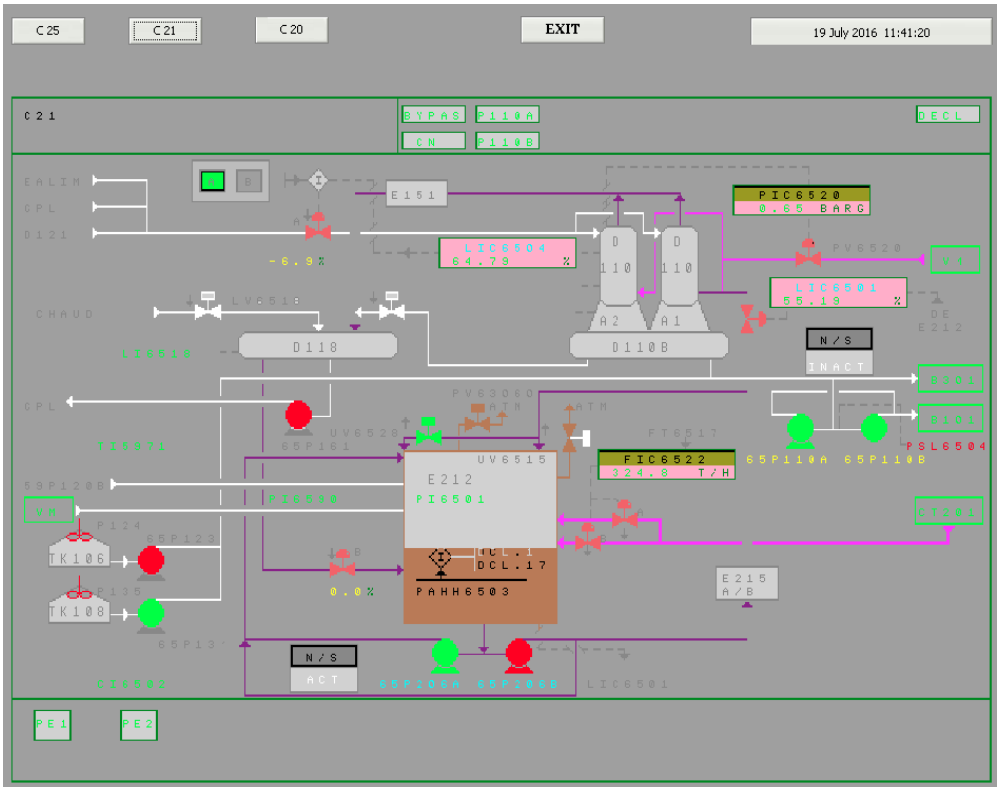

Fig. 12 Graphical simulation of the water supply

VII. Conclusion

The goal of our work is to use a rigorous approach for a good representation and implementation of a complex industrial system. This approach is based on the methodical approach AGR such as defined in AALAADIN and validated by the Multi-Agents platform MADKIT.

The advantage of this approach consists of its adaptability along with the possibilities of extension. Moreover, the decomposition into subsystems reduces significantly the complexity of the items being implemented and therefore, it allows a great modularity and a best legibility to the system.

References

[1] P. A. Fishwick, "Simulation Model Design and Execution. Building Digital Worlds," *Prentice Hall*,1995.

[2] M. Sonnessa, "Modelling and simulation of complex systems (doctoral dissertation)," PhD Thesis in "Cultura e impresa," *University of Torino, Italy*, 2004.

[3] R. E. Shannon, "Simulation modeling and methodology," *Proceedings of the 76 bicentennial conference on Winter simulation*, 1976, pp. 9-15.

[4] C. Oussalah, "Modèles hiérachisés multi-vues pour le support de raisonnement dans les domaines techniques," *Technical report*, 1988.

[5] R.G. Ingalls, "Introduction to simulation," *Proceedings of the 33nd conference on Winter simulation*. IEEE Computer Society, 2001, pp. 7-16.

[6] R. E. Shannon, "Introduction to the art and science of simulation," *Proceedings of the 30th conference on Winter simulation*. IEEE Computer Society Press, 1998, pp. 7-14.

[7] P. A. Fishwick, "Computer simulation: growth through extension," *Transactions of the Society for Computer Simulation International, 14(1)*, 1997, pp. 13–23.

[8] H. Vangheluwe, "Foundations of modelling and simulation of complex systems," *Electronic communication of the EASST*, 10: Graph Transformation and Visual Modeling Techniques, 2008.

[9] A. Simon, "The sciences of the artificial," *Cambridge (MA)*: MIT Press, 1969.

[10] L. V. Bertalanffy, "Théorie générale des systèmes," *Dunod*, 1968.

[11] O. Gutknecht, and J. Ferber, "The MadKit agent plateforme architecture," *Laboratoire d'Informatique, Robotique et Microélectronique de Montpellier*, 2000.

[12] J. Ferber, and O. Gutknecht, "Aalaadin: A meta-model for the analysis and design of organizations in multi-agent systems," *ICMAS (International Conference on Multi-Agent Systems)*, Paris, Y. Demazeau (ed), IEEE Press, 1998, pp. 128-135.

[13] B. Chaib-Draa, "Agent et Système Multi – Agents," *Université Laval, Quebec (Canada)*, 1999.

[14] J. Ferber, "Les systèmes multi-agents: vers une intelligence collective. Informatique," *intelligenceArtificielle*. Intereditions Paris, 1995.

[15] J. Ferber, "Les Systèmes Multi-Agents: Un Aperçu Général," *Revue Technique et Science Informatiques*, Hermes-Lavoisier, 1997.

[16] F. Michel, J. Ferber and A. Drogoul, "Multi-Agent Systems and Simulation: A survey from the agent's community perspective, " *Revue Technique et Science Informatiques*, Multi-Agent systems: simulation and application edited by A. M. Uhrmacher, D. Weyns– CRC Press-Taylor and Francis Group , 2009, pp. 3-52.

[17] N. Seddari, M. Redjimi and S. Boukelkoul, "Using of DEVS and MAS Tools for Modeling and Simulation of an Industrial Steam Generator," *CIT 22*, 2014, pp.171–189.doi:10.2498/cit.1002348.

[18] A. H. Bond and L. Gasser, "Readings in Distributed Artificial Intelligence," *Morgan Kaufmann Publishers*: San Mateo, CA, 1988.

[19] S. Franklin and A. Graesser, "Is it an agent, or just a program?: A taxonomy for autonomous agents," *In J. P. Mueller, M. Wooldridge, and N. R. Jennings, editors, Intelligent Agents III: Theories, Architectures, and Languages (LNAI Volume 1193)*, 1997, pp 21-35.

[20] C. A. Iglesias, M. Garijo, J. C. Gonzàlez and J. R. Velasco, "Analysis ans design of multi-agent systems using mas-commonkads," *In AAAI'97 Workshop on Agent Theories, Architectures and Languages, Providence, RI*, 1997.

[21] J. Ferber, O. Gutknecht,and F. Michel, "From agents to organizations: An organizational view of multi-agent systems," *In Paolo Giorgini, Jörg P. Müller, and James Odell, editors, AOSE*, volume 2935 of Lecture Notes in Computer Science, 2003,pp. 214–230.

[22] N. Seddari, M. Redjimi and L. Benoudina, "Operational approach for

modeling and simulation of an industrial process," *In IEEE International Conference on Computer Application Technology (ICCAT)*, 2013.DOI: 10.1109/ICCAT.2013.6522031.

[23] N. Seddari and M. Redjimi, "Multi-Agent Modeling of a Complex System," *In IEEE 3rd International Conference on Information Technology and e_Services (ICITeS),* Sousse – Tunisia, 2013.DOI: 10.1109/ICITeS.2013.6624072.

[24] F. J. Ferber and O. Gutknecht, "Generic Simulation Tools Based on MAS Organization*" LIRMM Laboratoire d'Informatique, Robotique et Micro-électronique de Montpellier,* 2001.

[25] J. Ferber, O. Gutknecht and F. Michel, "From agents to organizations: An organizational view of multi-agent systems," *In Paolo Giorgini, Jörg*P. Müller, and James Odell, editors, AOSE, volume 2935 of Lecture Notes in Computer Science, 2013, pp. 214–230.