A Multiagent Architecture for Supervisory and Control System

Ivan Rizzo Guilherme[1], Rafael Pedrosanto[1], Alex F. Teixeira[2], Celso K. Morooka[2], Carles Sierra[3]

[1] São Paulo State University, IGCE, DEMAC, Av. 24A 1511,
13506700 Rio Claro, Brazil
ivan@rc.unesp.br
[2] The State University of Campinas, FEM, DEP,
Campinas, Brazil
{alex,morooka}@dep.fem.unicamp.br
[3] IIIA-CSIC, Campus UAB,
09193 Bellaterra, Barcelona, Spain
sierra@iiia.csic.es

Abstract

Supervising and controlling the many processes involved in petroleum production is both dangerous and complex. Herein, we propose a multiagent supervisory and control system for handle continuous processes like those in chemical and petroleum industries In its architeture, there are agents responsible for managing data production and analysis, and also the production equipments. Fuzzy controllers were used as control agents. The application of a fuzzy control system to managing an off-shore installation for petroleum production onto a submarine separation process is described.

1. Introduction

In industries, the ample use of controller devices and sensors, in addition to the high interconnectivity between the equipments, generates such an environment that ends up yielding large quantities of information. In order to analyse data and supervise the processes in this scenario, complex computer systems are necessary, which are composed of several interacting parts. Designing these supervisory systems through the conventional techniques for software development is a hard task.

Studies with multiagent systems are regarded as the most promising ground for developing new tools to approach complex issues in Computer Science. Multiagents have yet provided many solutions for computing and engineering fields ([1]).

Research groups have been carrying out studies with multiagent systems ([2,3,4]), and others, are surveying methods and technologies based on the agents concepts for application in analysing, designing and implementing systems that are used in many fields by companies, but mainly in e-commerce, management and supervision of industrial processes. Theses agent-based methodologies enable the development of flexible, autonomous, and interactive components.

A traditional institution can be viewed as a set of artificial restrictions regulating interactions between agents. A good classical representation of this idea is that of an auction house with well-defined rules – as in any English auction. Some agents would be playing auctioneers, some will be salesmen, and others, buyers. In computer-mediated interactions, the Electronic Institution (EI) stands as a similar restricted virtual environment, wherein special interactions among the participants take place ([3]). Software agents are a keystone concept in EI, and are moulded as to embody this coordinating mechanism that regulates the interactions among software agents playing different business roles.

Within this field of automation and control, the several current approaches have been aiming at developing multiagent systems for managing and handling discrete processes. There is, however, a lack of standards for the methodologies in designing the multiagent softwares, which handle continuous processes like those in chemical and petroleum industries ([5]).

978-0-7695-3514-2/08 $25.00 © 2008 IEEE
DOI 10.1109/CIMCA.2008.152

Offshore petroleum production depends on dynamic and complex structures manned by qualified professionals with various backgrounds, and it is very expensive. Any operational complications and process instabilities may halt the production, resulting in additional costs. Predicting these problems therefore becomes relevant to ensure continuous production, secure operational safety and prevent environmental damages. The assessment of system breakdowns and solving of specific issues should not be deliberately carried out without special background expertise, which is necessary to follow the sequence of measures of the company's operational standards. Starting or restarting the production line also demands special background, in order to correctly take the necessary steps.

The present study introduces a multiagent architecture designed for doing control and supervision. For designing it, several related studies were taken in account. Jennings ([5]) suggested a methodology for designing supervisory and control systems for discrete industrial processes – the Manufacture Systems. Other multiagent architectures were proposed by Bunch ([6]),), Gabbar ([7]) and Mařík ([8]).

This way, we herein present a multiagent architecture for doing supervision and control. The supervisory and control agents were modelled on concepts of an EI, and the control agents work under fuzzy controllers. Based on the proposed architecture, we describe the control agents responsible for controlling a separation process in an offshore petroleum production plant using VASPS (*Vertical Annular Separation and Pumping System*).

2. Multiagent architecture

Multiagent architecture here proposed is designed to handles continuous processes like those in chemical and petroleum industries.

The proposed architecture (Figure 1) is composed of two types of agents: control agents and supervisory agents, described as follows:

- *Control agents*: agents executing the control of systems and of the components of the production plant. The necessary expertise for controlling the equipments operation was embedded into these agents through concepts of Fuzzy Sets Theory.
- *Supervisory agents*: agents that can integrate knowledge and data from different repositories with information provided by the control agents, with the objective of solving more complex problems. This type of agents can also exchange information with control agents to solve eventual complications. In the repositories are ready as ontologies related to matters of operational safety, diagnosing equipments malfunction, initiating and interrupting the production line, and others.

For modelling the multiagent architecture, we used a framework– the Electronic Institutions Integrated Development Environment, or EIDE ([3]). This way, control agents were defined as parts of an EI.

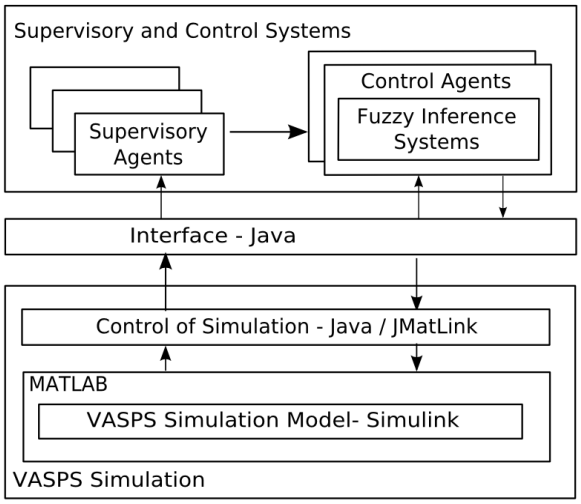

Figure 1: Diagram of the Simulator Integration in Java (EIDE) of the process

2.1 Modelling the Multiagent System

The modelling of a multiagent system follows concepts of an EI and uses available tools from EIDE. Hence, we will present the modelling of the proposed system as an EI, along with the supervisory agents.

Using the Islander tool from EIDE, we modelled a system composed of four parts: the *Dialogical Framework,* the *Performing Structure*, the *Control Scene* and the *Supervision Scene*.

The first one created was the *Dialogical Framework.* In this structure, the roles which agents can assume in the EI were specified, and three of them are cited below:

(a) *Agent*: basic role from which both others originate;

(b) *Controller*: role assumed by agents in the control level of the system;

(c) *Supervisory*: role assumed by agents in the supervisory level of the system.

Once all essential roles were assigned, we created a *Performative Structure* (Figure 2) which generally delineates the multiagent system dynamics. This structure was composed of distinct scenes:

a) *Initial scene*: enables introducing or creating agents for the institution;

b)*Control scene*: hosts the controller level;

c) upervision scene: hosts the supervisory level;

d) Final scene: where output or destruction of agents takes place.

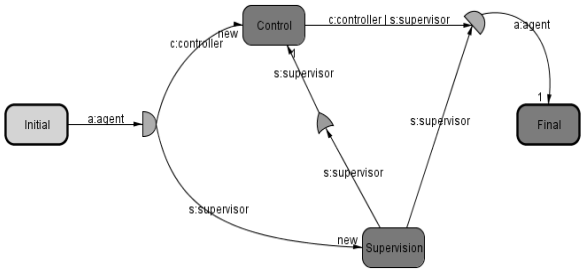

Figure 2. Representation of the *Performative Structure* of the proposed system.

In the *Performative Structure*, restrictions to the agents' actions are defined. Thus, every control agent (controllers) entering the EI is directly sent to the control scene, only leaving to be sent the final scene – during a system failure, for example. Supervisory agents proceed differently for, in spite of being sent to the supervision scene once entered the EI, for also they may be sent to the control scene along the process. This mobility is necessary, for whenever a supervisory agent has to share information with the control level, it has to be sent directly to where the control agents are. For example, an agent responsible for checking the VASPS pump temperature may need to report control agents on some instability.

Figures 3 and 4 respectively illustrate the Control and Supervisory scenes. Each of the two presented performative structures has three distinct states: S0 and S2, which respectively stand for initial and final states, except that only controllers can enter or leave the control scene and only supervisory agents can enter or leave the supervision scene. Note the signs (+) and (-) before roles' definitions: they stand for 'entering' and 'exiting'.

State S1 in both scenes is the state in which agents are performing their roles, although with some differences. Supervision scene's supervisor agents may leave the scene, i.e. duplicating themselves – useful whenever an agent needs to report the control level on problems, as above explained. In the control scene, a supervisory agent (in S1 state) can perform a 'stop and go' action, indicated by the sequence (+ -) followed by the role (–): this means that the agent entered a scene, performed an action and immediately left that scene.

Up to now, we have wholly defined the control scene without minding control aspects like equipment maintenance, planning or operational safety (these will be detailed in future studies). Thus, till now, only control agents were implemented in the project. For this, we designed a model using Matlab, which permits simulating production processes using VASPS (Figure 3). The integration of control agents into the process for simulation will be detailed in the following sections.

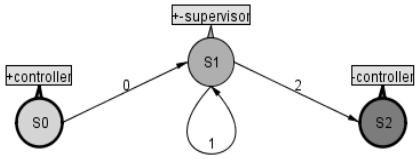

Figure 3. *Control Scene* of the proposed system.

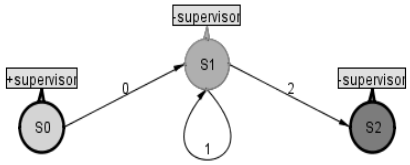

Figure 4. *Supervisory Scene* of the proposed system.

3. Offshore Separation System

Deep sea systems for production and separation are among the greatest technological challenges to oil companies. They enable reducing the use of platforms, considerably reducing operational costs, for instance.

VASPS (*Vertical Annular Separation and Pumping System*) is an underwater system for pumping and separating biphasic mixtures, composed of a separating vessel and a pump. The whole structure is assembled on the bottom of the sea by using a false well. Figure 5 presents a simplified scheme of the VASPS and its main parts. The incoming gas-liquid mixture is submitted to a triphasic separation, going through an expansion chamber, a propeller and ending in a pool ([9]). The liquid inside the pool is then pumped up to the platform by a submarine electric pump (SEP), while the gases simply rise due to an existing difference of pressure between the separator and the collecting vessel up on the surface. Both the liquid and the gases are then led to the platform by individual pipelines. There is a choke valve in the end of the pipeline, which, in association with the pump, regulates the outflow of the oil-water mixture.

Controlling the level of the liquid inside the pool is paramount, since if it exceeds a maximum set limit, the excess might invade the propeller area, affecting the process of separation. On the other hand, the pump could be damaged if the level goes below the minimum set limit.

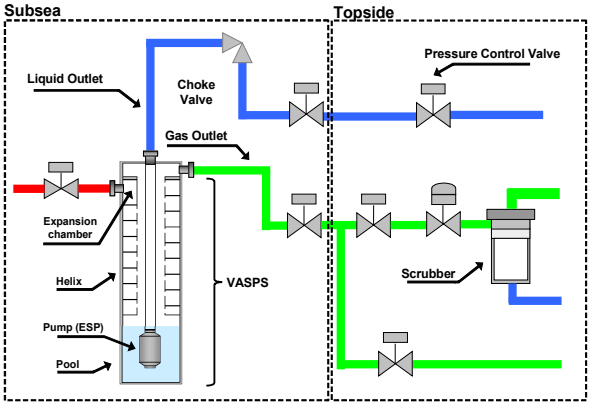

Figure 5. Simplified scheme of VASPS along with its main parts.

4. Architecture implementation

During the implementation phase of this architecture, we focused on developing the system's control agents. Figure 1 represents the system's architecture, having the process been designed in Simulink and the control agents work according to fuzzy control concepts.

The separation and pumping system is composed of two sections: a VASPS located at the bottom of the sea and a set of valves located at the surface platform (Figure 5). This VASPS Simulink-generated simulation model is an adaptation of the one proposed by Teixeira ([14]), having the control process been substituted by control agents.

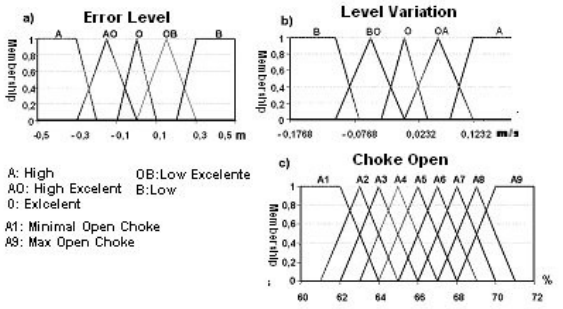

Figure 6. Fuzzy expertise of the control agent about a *choke* valve.

4.1 Control Agents

Two agents operating as fuzzy controllers are used in the control process – one regulates the *choke* valve, while another regulates the system pressure valve, thus keeping the pump's rotation steady during the process.

These control agents employ expertise expressed in fuzzy functions and rules. For each input variable in the controllers, five fuzzy sets were defined, while for output variable, nine fuzzy sets were defined. Two input variables were inserted in the *choke* valve control agent: Level Error and Level Variation (Figure 6). Based on their figures, the controller determines the valve aperture's value. Similarly, the agent responsible for controlling the pressure of the valve works with two variables: Pressure Error and Pressure Variation, which ultimately will define the control signal.

Control agents work by recovering values to be used as the system fuzzy inference input, by the interface of the simulated process. These agents issue a signal to control the process.

Control agents retrieve values of the pool level from the Simulink simulation, then calculating the level error and its derivate value. This composes their control signal, which contains a frequency increment to the pump's power supply. The control system has also a signal feedback from the controller's emission that works like a memory: it sums up the frequency variations of previous control signals to the present variation. This composes the control signal to be emitted to the pump.

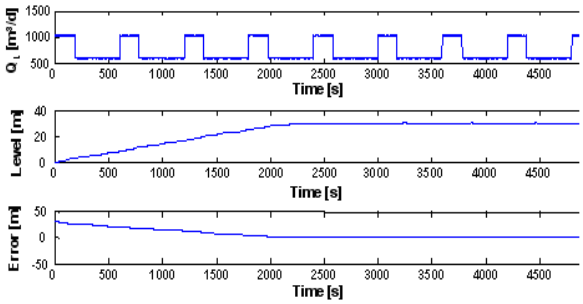

Figure 7: Test results: a) Incoming liquid flow b) Liquid level c) Level error

5. Integration between EIDE and the simulation environment

In this EIDE framework, there is a module (SIMDEI) that enables simulations with models designed in another module: the Islander. The simulation aims at evaluating the defined model. There are many tools for multiagent simulations, and SIMDEI employs the Repast tool ([10]). Nevertheless, Repast tool is only indicated for simulating multiagent systems in which the events in the processes are discrete and the occurring interactions are only between agents.

This way, in the targeted system environment, we end up having the agents and the simulation dynamics of the process; in it, agents must perceive the environment and react accordingly in a continuous manner. In order to do this, it was necessary defining simulation modules in which EIDE and the simulation environments were integrated, especially those permitting continuous process simulations. As EIDE is based on Java, it was necessary using simulation tools with Java-compatible mechanisms. The option adopted was Simulink ([11]), which has the advantage of permitting graphic representations of the processes to be simulated. Integrating EIDE (Java) to Simulink was made using JMatlink ([12]), as shown in Figure . JMatlink is a Java API that executes simulations made in Matlab or Simulink, based on Java. Moreover, JMatlink's resources permit accessing and following the simulated processes online ([13]).

6. Results

The tests were made using square wave signals representing the incoming liquid flow, as to simulate a significant alteration of the income conditions. This was made to ensure that the controller correctly reacts to great operational oscillations. The minimum simulated limit was set to 600 m^3/d while the maximum limit was set to 1030 m^3/d with peaks of 400s; the whole simulation lasted 5000s. The adopted set point level was 30m, measured from the base of the system. Results are presented in Figure 5.

Figure 7a illustrates the incoming liquid flow into the VASPS along time, which worked as an outer disturbance as to disorganize the process. Figure 7b illustrates the liquid level in the VASPS along time, in which the period of increase of 2200s can be noticed. Obtained stationary error was 0.85m, showing that the controller met its first goal.

6. Conclusion

The simulation results show that the developed pump control agent is reliable, satisfactorily working even under great operational variations, being capable of starting the system without human assistance. Both controlling strategies (pump or pressure control) can be used, and the mechanism enabling negotiation between the agents makes possible deciding the best strategy to be adopted at any given moment, setting the necessary adjustments according to the production flow.

Moreover, separating the decision-making mechanisms from the controllers added further flexibility to adopting the best control strategies.

The developed environment of fuzzy controllers in Java enables recycling projects developed in Matlab Toolbox.

Our next efforts will be on designing the Supervisory agents . These supervisory agents will interact with the control agents in the supervisory system.

Acknowledgments. This work was partially supported by CNPq (research project CT-Petro/MCT/CNPq 504863/2004-5 and MCT/CNPq 474039/2006-4)

7. References

[1] Klush, Matthias., Information agent tecnology for internet: A survey, Data & Knowledge Engineering, 36, (2001), 337-372.

[2] Jennings, Nicholas R., BUSSMANN, Stefan, Agent-Based Control-Systems. Why They Suited to Engineering Complex Systems. IEEE Control Systems Magazine, June 2003, pp. 61-73

[3] Sierra, Carles; Rodríguez-Aguilar, Juan Antonio; Noriega, Pablo, Esteva, Marc; Arcos, Josep Lluís; Engineering multiagent systems as electronic institutions. European Journal for the Informatics Professional, V(4), August 2004.

[4] Garcia, Alessandro, Objetos e Agentes: Uma Abordagem Orientada a Aspectos. Departamento de Informática, 2004. 298p. Tese de Doutorado - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

[5] Jennings, Nicholas R., BUSSMANN, Stefan, WOOLDRIDGE, M. Multiagent Systems for Manufacturing Control, Spring Verlag, 2004, pp. 288.

[6] Bunch, Larry; Breedy, Maggie; Bradshaw, Jeffrey M.; Carvalho, Marco; Suri, Niranjan; Uszok, Andrzej; Hansen, Jack; Pechoucek, Michal; Marik, Vladimir; (2004) Software Agents for Process Monitoring and Notification, ACM Symposium on Applied Computing.

[7] Gabbar, Hossam A.; Shinohara, Shintaro; Shimada, Yukiyasu; Suzuki, Kazuhiko; (2003), Experiment on distributed dynamic simulation for safety design of chemical plants, Simulation Modelling Practice and Theory,11, 109–123

[8] Mařík, Vladimír; Vrba, Pavel; Hall; Ken H.; Maturana, Francisco P.; Rockwell Automation Agents for Manufacturing, AAMAS'05, July, 2005, Utrecht, Netherlands.

[9] Benedetti, M., Villa, M. Field Tests on VASPS Separation and Pumping System. OTC 8449, 1997.

[10] Collier, N., 2003. REPAST: an extensible framework for agent simulation. http://repast.sourceforge.net

[11] Simulink. http://www.mathworks.com/products/simulink/

[12] Müller, S.: 2005. JMatLink library. http://jmatlink.sourceforge.net/

[13] Müller, Stefan; Waller, Heinz; (1999), Efficient Integration of Real-Time Hardware and Web-Based Services into MATLAB, Proceeding of 11th EUROPEAN SIMULATION SYMPOSIUM AND EXHIBITION, Germany

[14] Teixeira, Alex F., Mendes, José Ricardo P., Guilherme, Ivan R., MOROOKA, Celso K., ESTEVAM, Valdir, Um Controlador Fuzzy para o Sistema de Separação e Bombeamento Submarino – VASPS, Anais da Rio Oil & Gas Expo and Conference 2004.