A Methodology for Agent-Oriented Systems Development Applied in Oil Industry

Liviu IONIȚĂ, Irina IONIȚĂ

Petroleum-Gas University of Ploieşti, Blvd. Bucureşti, no. 39, Ploieşti, 100680, Romania,
lionita@gmail.com;tirinelle@yahoo.com

Abstract: Nowadays, the oil and gas industry faces a rapid evolution which involves the use of a wide specialized software based capabilities to achieve its business targets. Software and information technology represent the framework for a variety of business processes such as exploration, well construction, production optimization and operations. Complex oil and gas facilities gather key engineering and related disciplines needed to analyze, design, engineer, and operate the facility across its entire lifecycle. A solution that is emerging today to assist the processes developed within an industrial plant (e.g. gas-oil separation plant) is represented by intelligent agents. Due to their capabilities (reactivity, social ability, mobility, veracity, rationality, and learning/adaptation) agents can successful work together to solve complex and distributed problems associated to production, storing, transport and processing the petroleum products. After a careful analysis of agent-based methodologies, the authors of the current paper have chosen ZEUS methodology for development of a multi-agents system applied in oil industry. The research work consists in: defining of agents, the inter-agent communication, associating the roles of agents, implementing, testing and evaluating the designed multi-agent system.

Keywords: intelligent agent, multi-agent system, agent-oriented methodology, gas-oil separator

1. Introduction

In the last decades, many complex and distributed software systems, including process control system, diagnosis system, and modelling have used agent-oriented technologies (AOT). These new technology provides a new approach that aims at supporting the whole software development process (analysis, design, and implementation). The goal of AOT is to handle all phases and to offer a level of abstraction adequate to the problem to be solved, using a single, uniform concept, namely that of agents.

Agents are defined as autonomous entities, with cooperating and coordinating capacities, able to adapt to the new environment conditions and act together for accomplish a global objective. Due to these considerations, they have provided a path to build more robust intelligent applications from a different point of view.

AOT represent a natural extension to object-oriented techniques (OOT). In terms of OOT, agents can be seen as active objects. There are also main differences between objects and agents, as stated by [20]: the description of the internal state of an agent (by using notions like beliefs, goals, intentions etc.) and characterization of communication (by description of message types and the structuring of messages into protocols).

Over the past few years, researchers in various domains (computer science, information technology, engineering etc.) have worked together and have been several attempts at creating tools and methodologies for building multi-agent systems (MAS).

Although more methods and approaches have been proposed for this purpose, none of these methods have been accepted as a standard. The heterogeneous environment, the evolution of events, the probability of unexpected events occurrence, the difficulty to trace the system evolution involve producing a gap between agent oriented methods and the modelling needs of agent-based systems. A drawback of agent oriented software engineering methodologies, resulted from many discussions and research works presented in literature, is the lack of agreement on how to identify roles in the analysis phase and how to identify agent types in the design phase [22, 7].

This paper presents a methodology for building an agent-oriented system applied in oil industry. After a careful study, the authors have chosen the ZEUS methodology for multi-agent system development. MAS-GOSP is the proposed system that maps the processes of a Gas-Oil Separation Plant (GOSP) and consists in an agents' society with various specific assigned roles.

The article is organized as follows: section 2 gives the related work regarding agent-oriented methodology, section 3 contains the MAS-GOSP architecture and its functionalities, section 4 presents the experiments and the

results and the final section contains the conclusion and the future work.

2. Related Work

The agent technology has evolved rapidly along with a growing number of agent architectures, theories and languages and became one of the most important and active area of research and development. Agent oriented software engineering (AOSE) has numerous applications in various domains such as information management, air traffic control, electronic commerce, business process management, industry etc. A growing number of agent-based software engineering methodologies have been proposed in recent years in order to provide models, methods, tools and techniques for development of software systems in a systematic way [23].

The general template of an agent-oriented methodology consists in three submodels: the *agent model*, the *organizational model* and the *cooperation model* [2]. The agent model contains agents and their internal structure, described in terms of goals, plans and beliefs. The organizational model specifies the relationships between agents and agent types. Also here are mentioned the relationships among agents based on their assigned roles in organizations. The cooperation model describes the interaction among agents in a detailed way.

In literature, several methodologies for analysing, designing, and building multi-agent systems are based on theory of artificial intelligence (AI) coming from knowledge engineering (KE). Other methods extends the object-oriented methodologies or combine them with knowledge-based methodologies. The Figure 1 shows the development of agent-oriented methodologies (AOM) and the influences of object-oriented methodologies (OOM) on agent-oriented methodology (AOM).

The Multiagent Systems Engineering (MaSE) methodology, presented in [8,24] admits the influences from research work of authors Kendall, Malkoun and Jiang [12], as well as an heredity from AAII [13], which was significantly affected by the OOM recognized as Object Modeling Technique (OMT) [19]. The Gaia methodology [25, 26] uses the concepts of OOM of Fusion [4]. Tropos represents an AO extension from Gaia methodology. The Rational Unified Process (RUP) [14], other OO approach, provides the basis for ADELFE [1] and also for Methodology for Engineering Systems of Software Agents (MESSAGE) [3]. The aim of MESSAGE is to extend existing methodologies

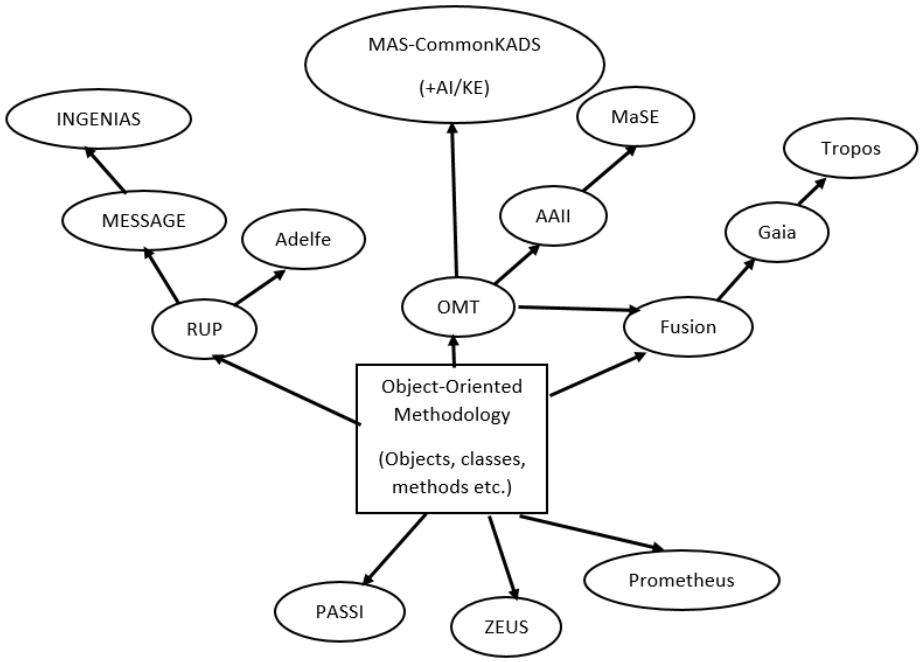

Figure 1. The influences of OOM on AOM (adapted from [11])

to allow them to support AOSE. UML is used for notation and also activity diagrams are generated. The analysis and design process in MESSAGE are based on the RUP [9].

INGENIAS [18] appears as an extension of MESSAGE, its concepts being also inherited from OOM. Prometheus [17], uses OO diagrams and concepts, even is not a descendent of OOM. PASSI [6] combines object-oriented concepts with MAS ideas, using UML description. ZEUS [16] is a methodology for building MAS, based on Java language, which represents an object-oriented programming language.

In the second part of this section, the authors shortly describe ZEUS methodology used for MAS implementation.

ZEUS [16] defines a multi-agent system design methodology, and automatically generates the executable source code of the user-defined agents. The ZEUS steps are: analysis, design, implementation and testing the system.

In the analysis phase, the main concern is to better understand the problem that will be solved. This methodology suggests using a specific method for analysis such as modelling roles [15].

An agent can have several roles and multiple responsibilities are associated to each role. A role model represents a template for the simplest solution of a problem. A role of an agent describes the position and the responsibilities' list of the agent in a certain context or role model. ZEUS vocabulary consists in associated roles of agents and the role models generated for multi-agent system description.

The first step of system design is to assign the roles identified at the previous step (analysis) to agents that form multi-agent system. The system developer translates the problem formulated initial in term of roles and responsibilities, in terms of agents and tasks or services. The problem is modelled as a multi-agent system concept. After this step, the declarative knowledge is modelled. This type of knowledge is used then by agents. In ZEUS, the concepts used are defined as facts and are categorized in: abstract and entity, in a hierarchical form. Each concept is characterized by a name and an attributes list that contains a name, a data type, constrains and an implicit value. The key concepts defined by the developer represent the ontology of the multi-agent system. The used terms will be found in the messages exchanged between agents. Their values may be modified as a result of task/service execution [15, 5].

Agent-based system implementation assumes the following sub-phases: creating ontology, creating each task agent (that implies agent definition, tasks description, agent organization and agent coordination), agent utilitarian configuring, task agent configuring and editing the Java source code). ZEUS provides a specific tool named Agent Generator responsible for making these steps. Also, a ZEUS agent called Visualizer helps the developer to test the designed multi-agent system. Its main role is debugging system, analysing messages passing, analysing agents' evolution and behaviour, analysing tasks execution and goals accomplishment [15, 5].

An important feature of ZEUS is represented by the embedded specialized editors (e.g. Ontology Editor, Task Description Editor, Organization Editor, Agent Definition Editor, Coordination Editor etc.) which essentially facilitate the identification and description of a set of agents, selecting agent functionality and inputting task and domain-related data. The output of the ZEUS methodology is a logical description of a set of agents and a set of tasks.

A ZEUS agent encapsulates the BDI model, meaning that Beliefs in ZEUS are translated to facts/beliefs, desires to goals, and intentions to commitments. The authors from [15, 5] consider that ZEUS methodology is more prescriptive and comprehensive than the BDI approach.

Having as starting point the above discussion regarding agent-oriented methodologies, in the following section, the authors of the current paper describe a multi-agent system architecture design for managing the activities of an industrial plant. In example given, a Gas-Oil Separation Plant (GOSP) is analyzed and modelled from an intelligent agent approach.

3. MAS-GOSP Architecture

A short description of GOSP

Before describing the multi-agent system architecture for GOSP, the authors shortly present the separation process in a three-phase separator.

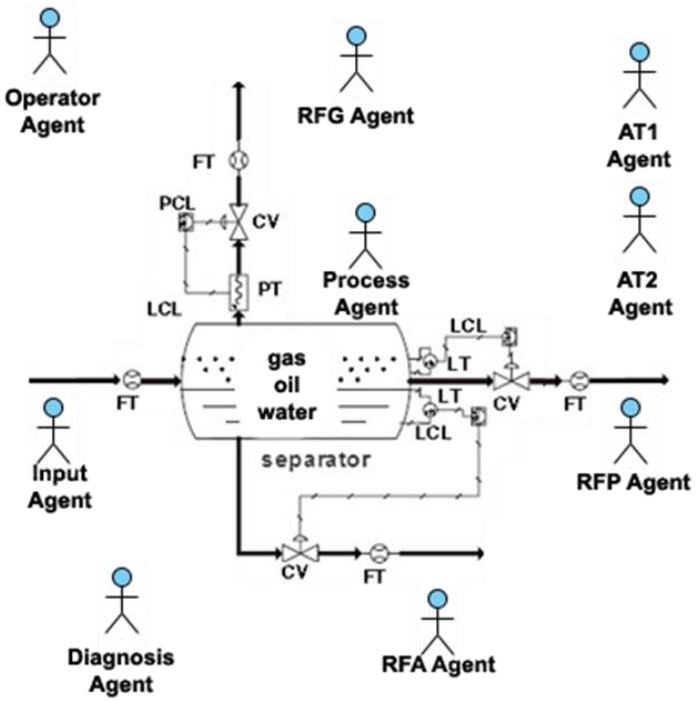

Figure 2. GOSP Agency

Inside of a three-phase separator the free water is separated and removed from the mixture of crude oil and water. Figure 2 contains a schematic representation of a three phase horizontal separator. The input of the separator is the fluid that hits an inlet diverter. At this moment, an initial gross separation of liquid and vapour is produced. In most designs, as is seen in literature [20] the inlet diverter contains a down comer that guides the liquid flow below the oil/water interface. The inlet mixture of oil and water is forced to mix with the water continuous phase in the bottom of the vessel and rise to the oil /water interface. In industrial domain terms, this process is known as "water-washing" [20]. The role of the inlet diverter is to assure that little gas is carried with the liquid. Supplementary, this device assures that the liquid is not injected above the gas /oil or oil /water interface, which would mix the liquid retained in the vessel. On the contrary, if the undesirable process occurred the control of the oil /water interface would became difficult. Some of the gas flows over the inlet diverter and then follow the horizontal line through the gravity settling section above the liquid. A consequence is that small drops of liquid mixed with gas are separated out by gravity and fall to the gas-liquid interface. The smallest drops are not easily separated in the gravity settling section and a new separation process is needed. As a result, before the gas leaves the vessel it passes through a coalescing section or mist extractor to coalesce and remove them [20].

Monitoring this complex process of separation is not an easy task. Automation devices such as transmitters, actuators, control valves, and controllers are used according to a control scheme, with several control loops in order to maintain the proper state of the GOSP.

In the next section, a gas-oil separator is modelled through agent-oriented approach, resulting a multi-agent system named MAS-GOSP. A detailed description of the proposed system and how it's working is given below.

System architecture

The three-phase separator is equipped with sensors (for pressure, temperature and level measuring), control valves, PID controllers and a "radiator" in the form of coil used for heating the mixture. The heater is connected to a boiler providing heating agent. Since the boiler serves several separators in the same park of separators, there is a second unit that supplies the heating agent rapidly, if it's necessary.

The environment is common to all the agents. In the example given, the environment is represented by the GOSP where agents act and accomplish assigned services. The agents' environment has the following characteristics:

- The environment is inaccessible – the agents do not have access to the entire state of the environment;
- The environment is non-deterministic – permanently, the environment is influenced by the other agents;
- The environment is not episodic;
- The environment is dynamic, continuously changing during the separation process;
- The environment is continuously, whereas the indicators go through a continuous range of values.

The architecture proposed for MAS-GOSP consists in nine agents with specific roles: Operator Agent, Diagnosis Agent, Input Agent, Process Agent, RFG Agent, RFP Agent, RFA Agent, AT1 Agent and AT2 Agent (Figure 2).

The agents that constitute the MAS for monitoring the GOSP have assigned several roles and responsibilities depicted below:

- Operator Agent – is responsible for providing all the information needed referring to the system functioning, to the human operator. This agent represents the interface between system and human operator, located to the control room.
- Input Agent – this agent has the role to monitor the inlet flow of mixture;
- Process Agent – this agent is responsible with the separation of fluid extracted from oil fields in three phases: gas, oil and water reservoir, through a fluid process heating using hot water;
- RFA Agent – this agent controls the water phase obtained after separation process;
- RFP Agent – the role of this agent is to control the oil phase obtained after separation process;
- RFG Agent – this agent controls the gas phase obtained after separation process;
- AT1 Agent – this agent is responsible with the heating unit HU1 that supplies the heat water necessary during the separation process;
- AT2 Agent – this agent has in charge the heating unit HU2, which supplies heating agent only if the HU1 does not provide, from different reasons (e.g. several faults occur during process), the heating agent quantity requested;
- Diagnostic Agent – this agent is responsible with system diagnosis, with subordinate agents RFA Agent, RFG Agent, RFP Agent and Process Agent.

The agents of MAS-GOSP form the control plan in real time and interact and collaborate together to achieve the proposed objectives. The planning process is characterized by a distributed and cooperative planning. Each agent creates its own plans and adapts them to the other agents' plans during the negotiation.

These agents proposed by the authors will be added utilitarian agents needed for a correct functioning of the MAS in ZEUS environment (ANS Agent, Facilitator and Visualizer).

Considering the system requirements and constrains, the initial analysis generate the following agent acquaintance diagram, presented in figure 3.

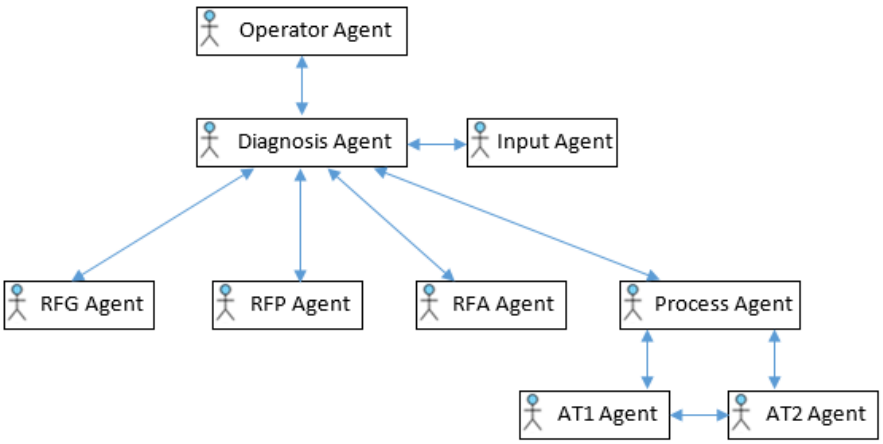

Figure 3. Agent acquaintance diagram

Each agent has assigned one or multiple roles, as is presented in the Table 1.

Table 1. The correspondent roles for MAS agents

Agent Name	Roles
Operator Agent	System data and alarm presentation to human operator
Diagnosis Agent	Alarm generation on the base of critical situation detection by agents: Input Agent, RFG Agent, RFP Agent, RFA Agent; inputs/output system presentation (input mixture flow, output gas flow, output oil flow and output water flow)
Process Agent	Heating mixture for separation in three phases (gas, oil, water) (initiator for AT1 Agent and AT2 Agent)
Input Agent	Three-phase Separator supply, input mixture flow measuring, critical situation detection (e.g. measured values outside the range of values)
AT1 Agent	Process supply with heating agent (Respondent to Process Agent, initiator for AT2 Agent)
AT2 Agent	Process and AT1 Agent supply with heating agent (Respondent to AT1 Agent and to Process Agent)
RFG Agent	Gas pressure measuring inside the separator, pressure control, critical situation detection (e.g. measured values outside the range of values), output gas flow measuring
RFP Agent	Oil level measuring inside the separator, level control, critical situation detection (e.g. measured values outside the range of values), output oil flow measuring
RFA Agent	Water level measuring inside the separator, level control, critical situation detection (e.g. measured values outside the range of values), output water flow measuring

The agent types of a MAS-GOSP are defined by considering the roles and scenarios of the system requirements. The agents should be evaluated against the criteria of coupling and cohesion, in order to establish the data needed by the different roles.

Table 2 contains a description of agents is given in form of Percepts, Actions, Goals and Environment known as PAGE description.

4. Experiments and Results

The authors present the following evaluation scenario, where only three agents are implemented in ZEUS framework: Process Agent, AT1 Agent and AT2 Agent. The formulation of scenario is: we assume that Process Agent requires heating agent from AT1 Agent and AT2 Agent. The availability for agents to accomplish this objective is different. The authors consider the case that AT1 Agent has a decrease availability (e.g. Availability=1), and AT2 Agent has an increase availability (e.g. Availability=10).

Figure 4 presents the graphical interface of the agents and the results of a negotiation.

During negotiation, the agents communicate and message change in order to achieve the goal. As a result, AT1 Agent is refused and AT2 Agent provides the heating agent for Process Agent.

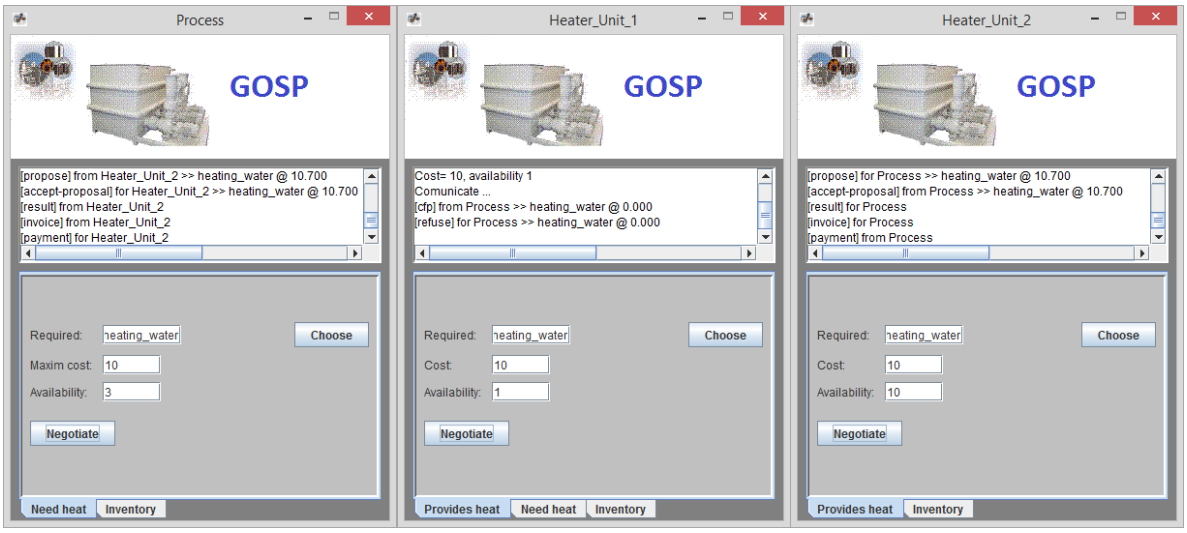

Figure 4. MAS-GOSP GUI

Table 2. The correspondent roles for MAS agents

Agent Name	Precepts	Actions	Goals	Environment
Input Agent	Measured values of input flow in the separator	Measures oil-well fluid, detects critical situation	Assures the optimal flow for a correct mixture separation	Gas-Oil Separation Plan
Process Agent	Measured values of mixture temperature	Negotiates with AT1 Agent and AT2 Agent to obtain heating agent	Heats the mixture for separation in three phases (gas, oil and water)	Gas-Oil Separation Plan
AT1 Agent	The need for heating agent (liquid units)	Negotiates with Process Agent for heating agent supply and negotiates with AT2 Agent for heating agent supplementation in case of unavailability /failure of AT1 Agent	Provides heating agent for Process Agent	Gas-Oil Separation Plan
AT2 Agent	The need for heating agent (liquid units)	Negotiates with Process Agent for heating agent supply and negotiates with AT1 Agent for providing supplementary heating agent	Provides heating agent for Process Agent and for AT1 Agent, only if is necessary	Gas-Oil Separation Plan
Operator Agent	Data offered by Diagnosis Agent	Represents and assists the human operator	Offers the information to the human operator regarding the global state of system operation and presents the alarms	Gas-Oil Separation Plan
Diagnosis Agent	Flow, level, pressure, temperature readings offered by Process Agent, Input Agent, RFG Agent, RFP Agent and RFA Agent	Transmits the system data to Operator Agent and identifies the possible failures occurred inside the system (generates alarms)	Maintains the system in the operating limits	Gas-Oil Separation Plan
RFG Agent	Gas pressure into separator	Opens, closes valve, adjusts gas pressure with pressure control loop, detects critical situation	Evacuates the gas phase from oil-well fluid due the separation process	Gas-Oil Separation Plan
RFP Agent	Oil level into separator	Opens, closes valve, adjusts oil level with level control loop, detects critical situation	Evacuates the oil phase from oil-well fluid due the separation process	Gas-Oil Separation Plan
RFA Agent	Water level into separator	Opens, closes valve, adjusts water level with level control loop, detects critical situation	Evacuates the water phase from oil-well fluid due the separation process	Gas-Oil Separation Plan

Figure 5 presents the information flow between the three implemented agents (Process Agent, AT1 Agent and AT2 Agent):

- Process Agent requires heating agent from AT1 Agent;

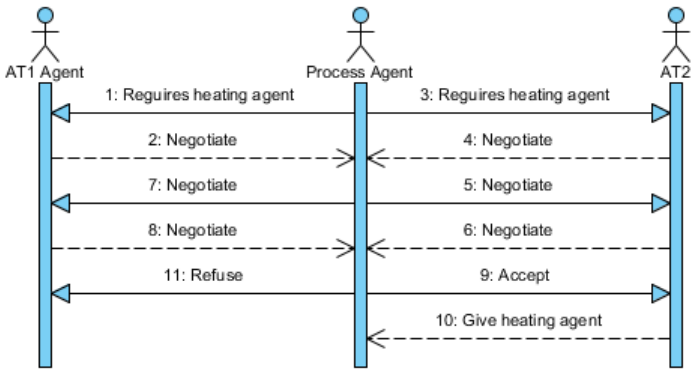

Figure 5. Negotiation

- Process Agent requires heating agent from AT2 Agent;
- AT2 Agent provides heating agent after negotiations to the Process Agent;
- AT1 Agent is refused (is unavailable because provides heating agent to other separator).

AT1 Agent uses FIPA Contract Net Protocol Manager coordinating protocol, and Growth Function strategy, as a strategy of initiator (tick "Initiator"). AT1 Agent also uses FIPA Contract Net Protocol Coordination Contractor, and Decay Function strategy, as the respondent's strategy (tick "Respondent").

In the example given, the authors consider Process Agent, AT1 Agent and AT2 Agent which form a subsystem of MAS. The entire agent-based architecture implementation represents authors' future work.

5. Conclusions

The current paper is based on the research work of the authors regarding the designing a multi-agent system applied in industrial field. MAS-GOSP consists in nine agents with specific roles which work together for goal achievement. For MAS implementation the authors uses ZEUS, an agent-oriented methodology.

In the example given, the environment of agents is a gas-oil separation plant (GOSP). The main contribution of the authors is represented by agent-oriented architecture that maps the real system on the oil field. Each agent has assigned a role in order to cover the entire functions of the GOSP. A PAGE description is used to better understand the agents' lifecycle in the MAS.

The research work presented in the paper consists in only three agents' implementation in ZEUS framework: Process Agent, AT1 Agent and AT2 Agent, that forms a subsystem of MAS. An evaluation scenario is formulated to describe the interaction between agents.

Future work will focus on implementing to the others agents of MAS so that MAS-GOSP will become a monitoring and diagnosis system.

REFERENCES

1. BERNON, C., M.-P. GLEIZES, G. PICARD, P. GLIZE, **The ADELFE Methodology for an Intranet System Design**, In P. Giorgini, Y. Lespérance, G. Wagner, & E. Yu (Eds.), Proceedings of Agent-Oriented Information Systems, AOIS-2002, AOIS.org, 2002, pp. 1-15.

2. BURMEISTER, B., **Models and Methodology for Agent-oriented Analysis and Design**, Working Notes of the KI 96.96-06, 1996, pp. 52.

3. CAIRE, G., W. COULIER, F. GARIJO, J. GOMEZ, J. PAVON, F. LEAL, P. CHAINHO, P. KEARNEY, J. STARK, R. EVANS, P. MASSONET, **Agent-oriented Analysis using MESSAGE/UML**, In M. Wooldridge, G. Wei, & P. Ciancarini (Eds.), Agent-oriented Software Engineering II LNCS 2222. Berlin: Springer-Verlag, 2001, pp. 119-135.

4. COLEMAN, D., P. ARNOLD, S. BODOFF, C. DOLLIN, H. GILCHRIST, **Object Oriented Development. The Fusion Method**, Englewood Cliffs, NJ: Prentice Hall, 1994

5. COLLIS, J., D. NDUMU, C. VAN BUSKRIK, **The ZEUS Technical Manual. Intelligent Systems Research Group**, BT Labs, British Telecommunications, 1999.

6. COSSENTINO, M., C. POTTS, **A CASE Tool supported Methodology for the Design of Multi-agent Systems**, In H. R. Ababnia & Y. Mun (Eds.), Proceedings of the 2002 International Conference on Software Engineering Research and Practice (SERP'02), Las Vegas, June 24-27, 2002, pp. 315-321.

7. DASTANI, M., J. HULSTIJN, F. DIGNUM, J. JULES, C. MEYER, **Issues in Multi-agent System Development**, ACM Press, International Conference on Autonomous Agents Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-agent Systems – Vol. 2, 2004, pp. 922-929.

8. DELOACH, S. A., **Multi-agent Systems Engineering: A Methodology and Language for Designing Agent Systems**, in Proceedings of the First International Bi-conference Workshop on Agent-Oriented Information Systems (AOIS '99), May 1, Seattle. AOIS.org, 1999.

9. EVANS, R., P. KEARNEY, G. CAIRE, F. GARIJO, J. GOMEZ SANZ, J. PAVON, P. MASSONET, **MESSAGE: Methodology for Engineering Systems of Software Agents**, EURESCOM Conference, EDIN, 0223-0907, 2001.

10. GARIJO, F. J., J. J. GOMEZ-SANZ, P. MASSONET, **The MESSAGE Methodology for Agent-oriented Analysis and Design,** Agent-Oriented Methodologies, vol. 8, 2005, pp. 203-235.

11. GIORGINI, P., B. HENDERSON-SELLER, **Agent-Oriented Methodology: An Introduction**, Idea Group Publishing, 2005

12. KENDALL, E. A., M. T. MALKOUN, C. JIANG, **A Methodology for Developing Agent based Systems for Enterprise Integration**, in P. Bernus & L. Nemes (Eds.), Modelling and Methodologies for Enterprise Integration. London: Chapman and Hall, 1996.

13. KINNY, D., M. GEORGEFF, M., A. RAO, **A Methodology and Modelling Technique for Systems of BDI Agents**, Technical Note 58, Australian Artificial Intelligence Institute, also published in Proceedings of Agents Breaking Away, the 7th European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW'96), Springer-Verlag, 1996, pp. 56-71.

14. KRUCHTEN, P., **The Rational Unified Process. An Introduction. Reading**, MA: Addison-Wesley, 1999.

15. NDUMU, D., J. C. COLLIS, H. S. NWANA, L. C. LEE, **The Zeus Agent Building Toolkit**, BT Technology Journal, vol. 16, issue 3, 1998.

16. NWANA, H., D. NDUMU, L. LEE, **ZEUS: An Advanced Tool-Kit for Engineering Distributed Multi-Agent Systems**, In: Proceedings of PAAM. 1998, pp. 377-392

17. PADGHAM, L., M. WINIKOFF, **Prometheus: A Methodology for Developing Intelligent Agents**, In F. Giunchiglia, J. Odell, G. Weiß (Eds.), Agent oriented Software Engineering III Proceedings of the Third International Workshop on Agent-Oriented Software Engineering (AAMAS'02), LNCS 2585, 2002, pp. 174-185.

18. PAVÓN, J., J. GOMEZ-SANZ, R. FUENTES, **The INGENIAS Methodology and Tools**, In B. Henderson-Sellers & P. Giorgini (Eds.), Agent-oriented Methodologies (Chapter 4). Hershey, PA: Idea Group, 2005.

19. RUMBAUGH, J., M. BLAHA, W. PREMERLANI, F. EDDY, W. LORENSEN, **Object-oriented Modelling and Design**, Englewood Cliffs, NJ: Prentice-Hall, 1991.

20. SAYDA, A. F., J. H. TAYLOR, **Modelling and Control of Three-Phase Gravity Separators in Oil Production Facilities**, American Control Conference, 2007. ACC '07, IEEE, 2007, pp. 4847-4853.

21. SHOHAM, Y., **Agent Oriented Programming**, Stanford University Technical Report STAN-CS-90-1335, Stanford, 1990.

22. STURM, A., D. DORI, O. SHEHORY, **Single-model Method for Specifying Multi-agent Systems**, ACM Press, 2003, pp. 121-128.

23. SUKHVIR, S., PRACHI, S. RICHA, **Evaluation of Agent Oriented Software Engineering (AOSE) Methodologies - A review**, International Journal of Latest Research in Science and Technology, Vol.1, Issue 2, ISSN (Online): 2278-5299, 2012, pp. 94-97.

24. WOOD, M., S. A. DELOACH, **An Overview of the Multi-agent Systems**

Engineering Methodology, in P. Ciancarini & M. Wooldridge (Eds.), Proceedings of the 1st International Workshop on Agent-Oriented Software Engineering (AOSE-2000), LNCS 1957, Springer-Verlag, 2000, pp. 207-222.

25. WOOLDRIDGE, M., N. R. JENNINGS, D. KINNY, **The Gaia Methodology for Agent-oriented Analysis and Design**, Journal Autonomous Agents and Multi-Agent Systems, vol. 3, 2000, pp. 285-312.

26. ZAMBONELLI, F., N. R. JENNINGS, M. WOOLDRIDGE, **Developing Multi-agent Systems: The Gaia Methodology**, In: ACM Transactions on Software Engineering and Methodology, vol. 12, issue 3, 2003, pp. 417-470.