Modeling and Implementation of a Reactive Multi-Agents Based Greenhouse Control System

Tuncay Aydoğan and Serap Ergün

***Abstract*—This paper presents development of a greenhouse model, which has a reactive multi-agent system, for the optimization of ambient agents such as temperature, light, humidity, and CO2 ratio and it has been validated on a prototype. The greenhouses are substantial agricultural structures whose internal ecosystem characteristics show continuous dynamic changes. Multi agent systems are computer systems that can autonomy control systems with dynamic environmental characteristics by interacting with other agents. In the conceptual design of the developed model, the state and control variables of the greenhouse automation have been modeled as reactive multi agents and the behavior, duty, message and input/output features of these agents have been identified. In its software design, the conceptual design features have been modeled with UML class diagrams, sequence diagrams and pseudo codes. The models have been applied to a prototype with JADE.NET. It was observed that the agents in PC user interface successfully and autonomy controlled the greenhouse by interacting with each other.**

***Index Terms*—Multi agent system; agent development platform; greenhouse control; JADE.NET.**

I. Introduction

The ever increasing world population and inadequate food production cause a nutrition problem worldwide. It is essential that agricultural areas be expanded and production be increased. However, the expansion of the production areas is almost impossible. The most common method for increasing production is to use greenhouses.

The greenhouse is a vegetative production structure that enables plants to maintain their all physiological activities such as growth and productivity in a consistent and the most productive way all year around. The most crucial agents that affect plants in a greenhouse are temperature, light, humidity, and CO_2 ratio in the air. It is difficult to say that the effects of these agents on the growth of a plant are independent of each other. Also known as agricultural air conditioning, heating, cooling, ventilation, shading, lighting, carbon dioxide fertilization and irrigation controls are in interaction with most of these agents at the same time. To keep the environmental conditions in a greenhouse at an optimum level under these interactions and relations is a complicated problem for greenhouse automations.

One of the modern approached used in industrial automation systems is intelligent agent technology. Basically, an intelligent agent is a software system that has flexible autonomous control ability, mostly consisting of more than one agent and is instable in a dynamic structure. It is observed that many applications have been developed together with intelligent agents in various fields such as mobile [1]-[4], internet [5]-[8], robotic [4], [9]-[11], control [9], [12]-[15], monitoring [16]-[19], semantic [20]-[25], and also greenhouses [26]-[30].

In this study, it is aimed to solve the problem of greenhouse air conditioning with intelligent agent architecture including to a reactive multi agent features. The solution consists of the stages of conceptual modelling, software modelling and implementation with a new agent design approach, and contains hardware and software components. In addition, the solution contains differences from the approaches in the literature in terms of agent design and modelling.

With conceptual modelling, it has been revealed that there is a relation between state and control variables of a greenhouse as a controlling system. The model has been designed as a reactive multi agent architecture comprising of eleven agents. The behaviours of agents, the relation among them have been put forth, and class diagrams, sequence diagrams and pseudo code haven been identified during the software modelling of the system. Lastly, the agents have been programmed at the platform of JADE.NET and the software control of the hardware components of the greenhouse automation has been implemented on a prototype.

II. Materials

A. Agent Terminologies: Architecture

The "agent" term has taken a place in different technologies such as artificial intelligence, operating systems, databases, and computer network literature. However, a single definition of an agent cannot found and also there are so many definitions of agent in [31]-[33]. Considering all definitions, the following definition may apply. An agent is substantially a special software component that provides autonomously an interoperable interface to a high-handed system. An agent system can be based on an individual agent working within an environment and they consist of multiple agents when interacting with its users is necessary. Herewith, these multi-agent systems (MAS) can model complex systems and acquaint the possibility of agents having mutual or contradictory goals. Agents may decide to cooperate for common benefit or may contest to serve their own interests. Likewise these agents can interact with each other directly or indirectly [34].

Manuscript received August 7, 2017; revised November 14, 2017.

The authors are with Suleyman Demirel University Faculty of Technology, Software Eng. Dept., Turkey (e-mail: tuncayaydogan@sdu.edu.tr).

doi: 10.18178/ijiee.2017.7.6.682

On account of this, an agent is autonomous, by virtue of operating without the direct intervention of humans/others and having control over its actions and internal state. An agent is social, by virtue of cooperating with humans/other agents to achieve its tasks. An agent is reactive, by virtue of perceiving its environment and responding in a timely fashion to changes that occur in the environment. And agent is proactive due to it does not simply move in response to its environment but is able to show aim-directed behaviour by taking initiative [32], [34].

Agent architectures are primary mechanisms underlying the autonomous components that backup effective behaviour in real-world, dynamic and open environments. Actually, initial efforts in the field of agent-based computing center on the development of intelligent agent architecture. Agent architectures can be divided into four main groups as logic based, reactive, BDI (Belief Desire Intention), and layered architecture [34], [35].

In this study, our multi-agent system's architecture based on reactive architecture that applies decision-making as a direct mapping of situation to action and is based on a stimulus-response mechanism triggered by sensor data. The advantage of using reactive agent architecture is that it performs better in dynamic environment.

B. Agent Terminologies: Communication and Programming Languages

Communication is one of the best important components of MASs. Agents can communicate with users, system resources and also if necessary, they can cooperate, collaborate, and confer. Especially, agents interact with each other by using agent communication languages that provide a separation between the communication acts and the content language. Mechanisms for the communication between agents support their social abilities. Different methods for agent communication have been developed, but KQML was the first communication language [36], [39]. The important components are MASs programming languages, platforms and development tools because of affecting the diffusion and using of agent technologies across different application areas.

MASs can be figure out by using any kind of programming language. Above all, object-oriented languages are considered a suitable means. The concept of agent is not too different from the concept of object. Some common properties of objects and agents like encapsulation, frequently, inheritance, and message passing. Nevertheless, agents differ from objects in several points as autonomous, capable of flexible behaviour, and own control [32], [40].

Agent-oriented programming languages focus on taking into account the main characteristics of MASs. Some attributes of agency such as beliefs, goals, plans, roles, and norms are supported by many agent-oriented programming languages. Several agent-oriented languages are available such as JaCaMo, JACK, JADE, 3APL and can be examined from [39]-[42].

In this study, we handle JADE.NET in order to agent modelling that is based on JADE (Java Agent Development framework) and most used in these days. JADE is an entirely distributed middleware system with a flexible infrastructure serving easy extension with add-on modules. It is written completely in Java, it benefits from the huge set of language characteristics, in this way offers a rich set of programming segregations allowing developers to construct JADE MASs with relatively minimal specialty in agent theory.

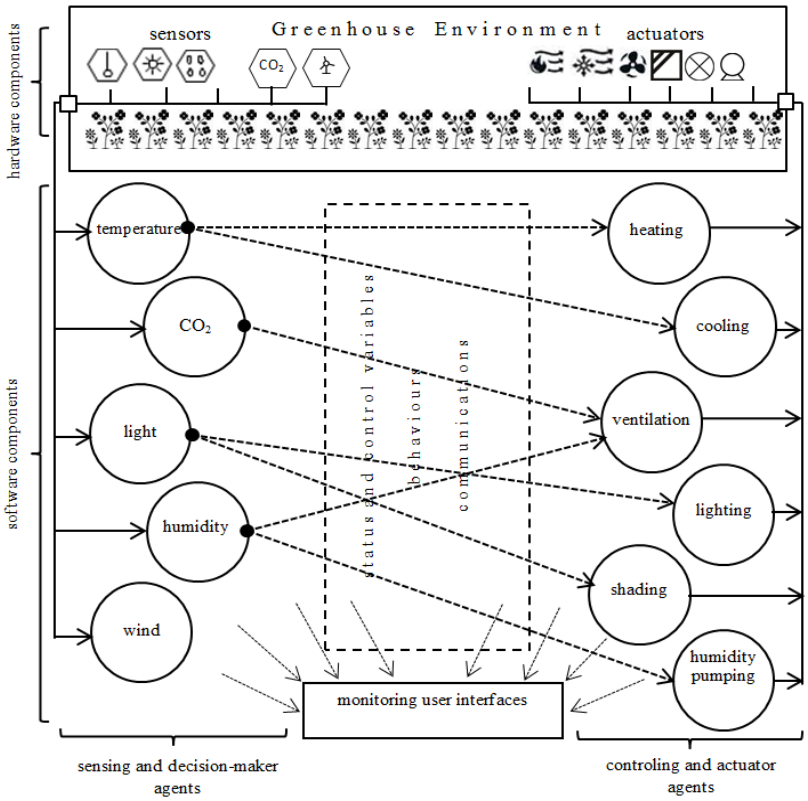

Fig. 1. Conceptual modeling of greenhouse multi agent system.

III. Methods

A. Conceptual Modeling

Within the framework of this study, internal temperature value, CO_2 ratio, quantity of light and humidity rate have been identified as state variables, and heating, cooling, ventilation, lighting and shading have been identified as control variables. The conceptual model originates from hardware and software components as seen in Fig. 1. Sensors and actuators are the hardware components of the system.

The controlling system takes the values of state variables from temperature, CO_2, light and humidity sensors. The actuator that will activate the heating, cooling, ventilation and shading is used with the aim of optimizing these values for the plants in the greenhouse. These actuators could be heaters, coolers, window systems and lighting systems in various technologies.

The agents are the software components of the system. In the greenhouse within this study, the agents which sense and decision-make are the temperature, CO_2, light and humidity agents have been designed as the controller agents such as humidity pumping, heating, cooling, ventilation, lighting and shading agents. The sensor agents maintain a perpetual agent life cycle by interacting with actuator agents and behaving according to the state values within the environment.

The duty of the temperature agent is to keep the internal temperature of the greenhouse at the targeted constant level in accordance with the value taken from temperature sensor. It makes the actuators of these agents to operate, when necessary, communicating with heating and cooling agents.

The humidity agent keeps the inner humidity rate of the greenhouse at a targeted constant level in accordance with the value taken from the humidity sensor. It makes the actuators of the humidity pump and ventilation to operate.

The duty of CO_2 is to provide collapsible windows by interacting with ventilation agent according to the value taken from CO_2 sensor when the inner CO_2 concentration of the greenhouse increases.

The light agent communicates with the lighting and shading agents by evaluating the lighting quantity according to the value taken from the light sensor and activates the actuators that will provide inner lighting and shading of the greenhouse.

The wind agent measures the external greenhouse wind speed and shows in the monitoring user interface.

Thanks to "monitoring user interfaces" unit, inside/ outside information of greenhouse, agents' behaviours and communication information can be viewed on a PC.

B. Software Modeling

In this study, the agents have been modelled with software tools and properties of JADE.NET platform. The duty, behaviour and communication relations of the agents can be seen in Table I. As shown on the table, the agents are in interaction with other agents, sensors or actuators.

TABLE I: Properties of Agents

Agent	*Duty*	*Behaviour Type*	*Messaging (to/from) agent*	*Input/Output*
temperature	*control to greenhouse inner temperature*	*TickerBehaviour CyclicBehaviour*	*to heating agent to cooling agent*	*from temperature sensor*
heating	*on/off to heater*	*CyclicBehaviour*	*from heating agent*	*to heater actuator*
cooling	*on/off to cooler*	*CyclicBehaviour*	*from heating agent*	*to cooler actuator*
CO_2	*control to greenhouse inner CO_2 ratio*	*TickerBehaviour CyclicBehaviour*	*to ventilating agent*	*from CO_2 sensor*
ventilation	*on/off to windows*	*CyclicBehaviour*	*from CO_2 agent*	*to windows actuator*
humidity	*control to greenhouse inner humidity ratio*	*TickerBehaviour CyclicBehaviour*	*to ventilating agent to humidity pumping agent*	*from humidity sensor*
humidity pumping	*on/off to humidity pump*	*CyclicBehaviour*	*from humidity agent*	*to humidity pumping actuator*
light	*control to greenhouse inner lighting*	*TickerBehaviour CyclicBehaviour*	*to lighting agent to shading agent*	*from light sensor*
lighting	*on/off to LEDs*	*CyclicBehaviour*	*from light agent*	*to LEDs actuator*
shading	*on/off to curtains*	*CyclicBehaviour*	*from light agent*	*to curtain actuator*
wind	*measure to wind speed*	*TickerBehaviour*	*-*	*from wind speed sensor to monitoring user interface*

For example, the temperature agent firstly reads the inner temperature with TickerBehaviours and then evaluated the situation and lastly, sends message to heating and cooling agents via CyclicBehaviour. The relevant agent that receives this message activates the actuator, which it is responsible for controlling.

An agent carries out its duties via behaviours. A behaviour is identified with the expansion of the class of jade.core.Behaviours.Behaviour by inheritance. In order for an agent to perform this behavior, it should be added to addBehaviour method as a parameter and should be called as addBehaviour(b). In the modelling, the behaviours of CyclicBehaviour and TickerBehaviour have been used [6].

The Cyclic behaviours are never terminated and are used in continuous duties. Whenever the behaviour is called back, processes of action method are repeated. TickerBehaviour is used for the duties to be implemented in a certain period of time. Whenever the time interval given to the method constructer as a parameter expired, the onTick method is added to repeat queue that is terminated [6].

Nowadays, the agents are modeled and identified with object oriented methodologies. The agent properties are identified as a class. UML (Unified Modeling Language) tools and diagrams have been adapted in a way that they can model the agents.

The aim of the class diagram is to describe the functions and properties of the class in a model. When the agent is regarded as an object and modeled as a class, the agents communicated by the other agents have been structured in a way that they have object characteristics, agent behaviours and functions.

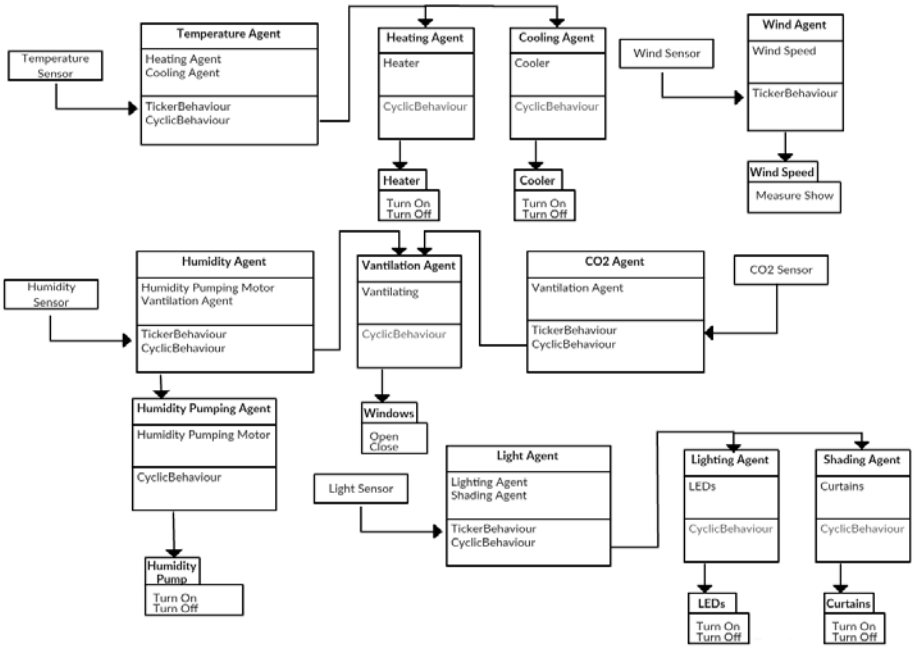

Fig. 2. UML class diagrams of agents.

The class diagram of the agents that built the greenhouse model has been shown in Figure 2. For example, the properties of temperature, heating and cooling agents have Ticker and Cyclic Behaviours. The agents and state/control variables in interaction with agent behaviours are associated with arrows.

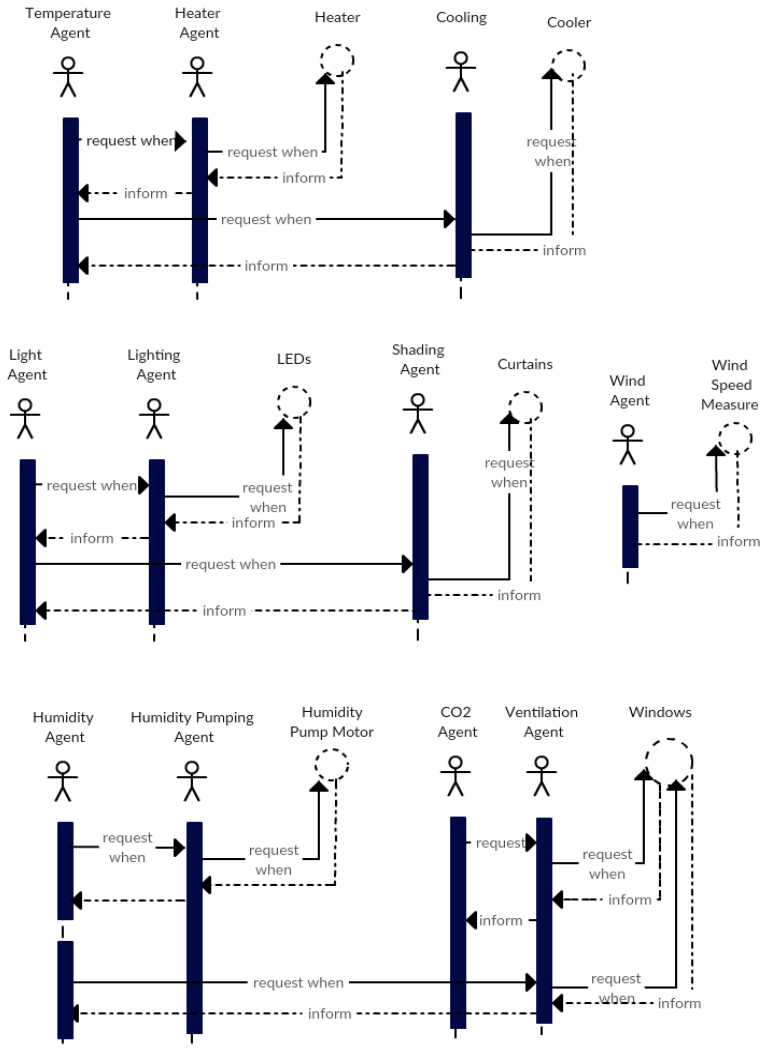

Fig. 3. UML sequence diagrams of agents.

A sequence diagram is an UML diagram that shows how objects operate with one another and in what order. A sequence diagram basically consists of objects, messages and time scales. The objects are structure blocks that are defined for carrying out certain duties as in UML diagrams. The objects in sequence diagram are physically read from top to bottom and from left to right. The dashed line under each object represents the timeline. The long and narrow rectangular on the dashed line showing the time flow represents the activities carried by that object in a certain period of time. The messages consist of the arrows drawn from the time line of an object to the time line of the other object. The communication between the objects and the commands they send to each other are represented with these arrows.

The sequence diagrams of the agents that construct the greenhouse model can be seen from Fig. 3.

For example; the temperature agent communicates with the heating agent for on/ off of heaters which is the task of the heating agent. The heating agent send a message to the heater for on/off the heaters, the heaters send a message of on/off information to the heating agent. As a next step, the temperature agent communicates with the cooling agent for on/ off of coolers which is the task of the heating agent. The heating agent send a message to the cooler for on/off the coolers, the heaters send a message of on/off information to the cooling agent.

Jade, JadeLeap has been added to the library files as references in order to model the properties and behaviours of an agent in Microsoft Visual Studio. Jade, jade.core.behaviours, jade.core.@event, jade.core.management, jade.domain.introspection, jade.lang.acl, jade.core, jade.lang.acl, jade.util.leap and namespaces have been provided after their identification to the compiler.

The pseudo codes of temperature agent's algorithm are given in Fig. 4.

Algorithm 1: TemperatureAgent
Input: Reading ***internaltemperature(inttemp)*** from *SerialPort1*, int ***desiredtemperature(destemp)***
Output: **SentMessage** to *HeatingAgent*
SentMessage to *CoolingAgent*

if inttemp<destemp **then**
set ACLMessage as **msg**
if msg!=null **then**
SentMessage by SerialPort "**1**" information to *HeatingAgent*
set *HeatingAgent* as *StateReady*
set *HeatingAgent* as *StateRunning*
while inttemp=destemp
turned on *Heaters by HeatingAgent*
then turned off *Heaters by HeatingAgent*
return *ReceivedMessage* to *TemperatureAgent*
end elseif inttemp>destemp
set ACLMessage as **msg**
if msg!=null **then**
SentMessage by SerialPort "**1**" information to *CoolingAgent*
set *CoolingAgent* as *StateReady*
set *CoolingAgent* as *StateRunning*
while inttemp=destemp
turned on *Fans by CoolingAgent*
then turned off *Fans by CoolingAgent*
return *ReceivedMessage* to *TemperatureAgent*
end

Fig. 4. Pseudo codes of the temperature agent.

C. Implementation

Temperature, humidity, CO_2, ventilation, lighting, lighting, shading, cooling, irrigation and wind agents are created for this study. The aim of the temperature agent; keeping constant internal temperature set point, heating, cooling, heating if necessary communicates with irrigation agent is to ensure that the introduction of irrigation and cooling agents. The internal humidity of the greenhouse is trying to keep in balance thanks to humidity agent. When the domestic greenhouse CO_2 concentration increased ventilation agent calling task is to ensure the opening and closing of windows by CO_2 agent. Light agent is responsible for examining whether the light inside the greenhouse lighting to ensure lighting and shading agents communicating performs open and close the curtains agents. To make the greenhouse temperature in irrigation work fulfils is the task of an irrigation agent and its duties in accordance with the message received from the agent. Wind agent shows the wind speed outside the greenhouse system.

```
namespace Greenhouse
{
    public class TemperatureAgent
    {
        ACLMessage incoming = TemperatureAgent.Equals(incoming, null);
        static ACLMessage incoming null;
        override public void setup()

        {
            addBehaviour(new CollectMsgsBeh());
            addBehaviour(new WaitBeh());
        }

        public class CollectMsgsBeh : CyclicBehaviour
            override public void action()
            {
                int count = 0;
                int inttemp;
                int wantedtemp;
                if (inttemp < wantedtemp)
                {
                    ACLMessage msg = new
jade.domain.introspection.ACLMessage(ACLMessage.PROPOSE);
                    msg.addReceiver(new AID("HeatingAgent", AID.ISLOCALNAME));
                    msg.setLanguage("Agent Language");
                    msg.setOntology("Heater On/Off");
                    SenderBehaviour();
                }
                if (inttemp > wantedtemp)
                {
                    ACLMessage msg = new
jade.domain.introspection.ACLMessage(ACLMessage.PROPOSE);
                    msg.addReceiver(new AID("CoolingAgent", AID.ISLOCALNAME));
                    msg.setLanguage("Agent Language ");
                    msg.setOntology("Cooler On/Off");
                    SenderBehaviour();
                }           }       }
        public class WaitBeh : TickerBehaviour
                    protected void onTick()
            {
                incoming = ReceivedMessage();
                if (incoming != null)
                {
                    longIncoming(incoming);
                    if (incoming.getAclRepresentation() != ACLMessage.Equals)
                    { incoming = null; }
                }    block(500);
            }
            private ACLMessage ReceivedMessage()
            {
                throw new NotImplementedException();
            }     }       }   }}
```

Fig. 5. JADE.NET codes for temperature agent.

For example; the temperature agent with TickerBehaviour assesses the situation by reading the temperature inside the greenhouse temperature sensor (Fig. 5). With CyclicBehaviour, it sends the messages to the relevant agent(s). The classes are set before, and then the parameters are defined for the purposes of each behaviour. ACL Message determined to send messages to related agents. The relevant agents taken the message activate the control elements within that mission.

The agents have been programmed according to the optimum temperature, humidity and CO_2 values of the tomatoes, peppers and eggplants in the periods of germination, seedling and post-planting during the implementation.

Two monitoring user interfaces have been designed for monitoring to the state/control variables of the greenhouse and the behaviours / messages of the agents.

The immediate data related to the state and control variables of the greenhouse in accordance with the plant chosen are monitored from the first interface in Fig. 6.

"Execute System" interface when the button is pressed, the information is transferred to the label reads from Serial Port. "Appropriate for production values" button is pressed plants selected from the combobox and according to the plant's production phase begins to provide the necessary conditions. The changes can be followed through colouring over the label. "Stop the System" button is pressed, the system would be terminated.

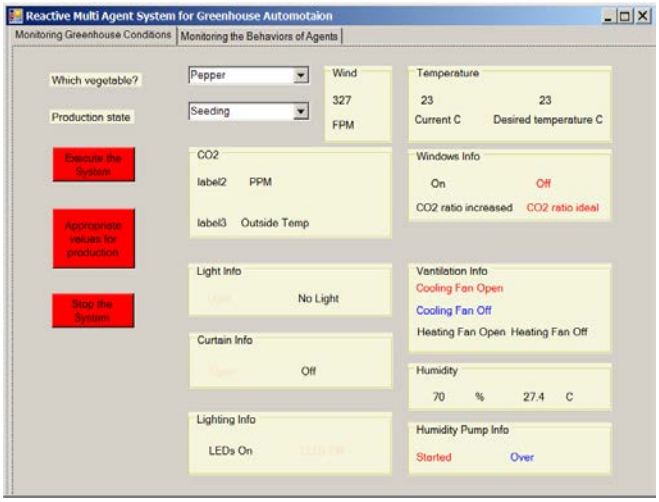

Fig. 6. Sensor / actuator monitoring of the situation of according to the plant selection and production phases.

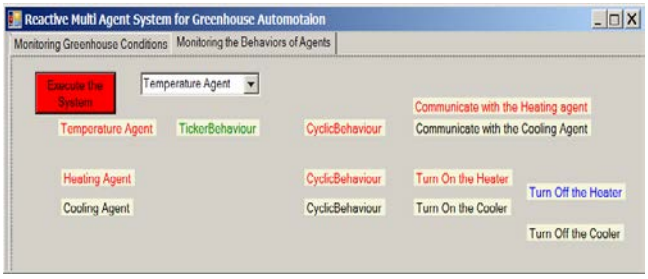

Fig. 7. Monitoring status of the agents.

In Fig. 7, behaviour of the agent selected and instant messaging activity is observed can be monitored. A view to the desired agent is selected from the combobox, and other relevant agents come together in the behaviour and actuator information. Behaviours have taken place and actuators are working as, actuator information comes with colouring on labels.

IV. Results and Conclusion

In this study, an automation system has been designed with intelligent agent technology that could keep the most important environmental agents such as temperature, light, humidity and carbon dioxide rate under control. The designed model comprises of the reactive multi agents like temperature, CO_2, light, humidity, humidity pumping, wind, heating, cooling, ventilation, lighting and shading. Some of the agents receive perceived states via the sensors and some of them control the actuators.

The paper describes the development and implementation of control system. Establishing necessary to provide active control of the greenhouse is aimed to be tested on a prototype and communicate with each other. The system is able to interact in both directions with the greenhouse, to read the parameter values of sensors and to control the actuators.

This study has demonstrated good quality of JADE.NET agent development platform applied to agent technology. Furthermore, the integration of multi agent technology with greenhouse is made possible thanks to definition of the Agent Behaviours.

The behavioural and messaging-related property of an agent has been implemented on a JADE.NET platform, being modelled with UML class diagrams, sequence diagrams and pseudo codes.

The contribution of the operation in hardware, thanks to the intelligent agent technology creates greenhouse prototype "sense-think-act" autonomously communicating unit of the environment (automatic) is shown the system can be developed.

Consequently, it has been seen that the designed model could control autonomously a greenhouse prototype where all agents are in interaction with each other.

V. Outlook and Future Works

Using the approach of this study, greenhouse, and automation/control firms can meet the technical challenges of multi-agents systems and build comprehensive and effective control systems. The most interesting property of our work is the synergy achieved between intelligent agent systems and control systems world in greenhouse-based application areas.

As a future work, we plan to implement our model in other multi-agents development platform by including the different agent technologies such as mobile agents. After that, studies are comparable to each other.

Acknowledgment

The authors would like to thank to Süleyman Demirel University Research Project Coordination Unit.

REFERENCES

[1] A. Brugués, S. Bromuri, J. Pegueroles, and M. Schumacher, "MAGPIE: Mobile computing with Agents and publish/subscribe for intelligent u-hEalthcare," *Swiss Medical Informatics*, vol. 31, 2015.

[2] F. A. Asnicar and C. Tasso, "ifWeb: A prototype of user model-based intelligent agent for document filtering and navigation in the world wide web," in *Proc. 6th International Conference on User Modeling*, 1997, pp. 2-5.

[3] T. Magedanz and T. Eckardt, "Mobile software agents: A new paradigm for telecommunications management," *Network Operations and Management Symposium*, IEEE, vol. 2, pp. 360-369, 1996.

[4] H. Zohoor and S. M. Khorsandijou, "Dynamic model of a mobile robot with long spatially flexible links," *Scientia Iranica*, Trans. B, vol. 16, no. 5, pp. 387-412, 2009.

[5] H. O. Al-Sakran, "Intelligent traffic information system based on integration of internet of things and agent technology," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 6, no. 2, pp. 37-43, 2015.

[6] F. Bellifemine, G. Caire, T. Trucco, G. Rimassa, and R. Mungenast, "Jade administrator's guide," *TILab*, 2003.

[7] O. Etzioni and D. S. Weld, "Intelligent agents on the internet: Fact, fiction, and forecast," *IEEE Expert*, vol. 10, no. 4, pp. 44-49, 1995.

[8] S. Wang, J. Wan, D. Zhang, D. Li, and C. Zhang, "Towards smart factory for industry 4.0: a self-organized multi-agent system with big data based feedback and coordination," *Computer Networks*, vol. 101, pp. 158-168, 2016.

[9] M. K. Habib, H. Asama, Y. Lshida, A. Matsumoto, and I. Endo, "Simulation environment for an autonomous and decentralized multi-agent robotic system," in *Proc. the 1992 lEEE/RSJ International Conference on Intelligent Robots and Systems*, 1992, vol. 3, pp. 1550-1557.

[10] P. Iñigo-Blasco, F. Diaz-del-Rio, M. C. Romero-Ternero, D. Cagigas-Muñiz, and S. Vicente-Diaz, "Robotics software frameworks for multi-agent robotic systems development," *Robotics and Autonomous Systems*, vol. 60, no. 6, pp. 803-821, 2012.

[11] J. Hou, "Dynamic energy consumption and emission modelling of container terminal based on multi agents," in *Proc. MATEC Web of Conferences*, vol. 124, 2017, EDP Sciences, p. 06001.

[12] J. M. Corchado, J. Bajo, Y. De Paz, and D. I. Tapia, "Intelligent environment for monitoring Alzheimer patients, agent technology for health care," *Decision Support Systems*, vol. 44, no. 2, pp. 382-396, 2008.

[13] M. Dastani, "Programming multi-agent systems," *The Knowledge Engineering Review*, vol. 30, no. 4, pp. 394-418, 2015.

[14] M. F. Zarandi, I. B. Turksen, S. M. Hoseini, S. Bastani, and A. Mohebi, "A fuzzy intelligent information agent architecture for supply chains," *Scientia Iranica*, vol. 15, no. 5, pp. 623-636, 2008.

[15] Y. Tang, X. Xing, H. R. Karimi, L. Kocarev, and J. Kurths, "Tracking control of networked multi-agent systems under new characterizations of impulses and its applications in robotic systems," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 2, pp. 1299-1307, 2016.

[16] S. S. Heragu, R. J. Graves, B. I. Kim, and A. St Onge, "Intelligent agent based framework for manufacturing systems control," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 32, no. 5, pp. 560-573, 2002.

[17] E. Fernández, C. M. Toledo, M. R. Galli, E. Salomone, and O. Chiotti, "Agent-based monitoring service for management of disruptive events in supply chains," *Computers in Industry*, vol. 70, pp. 89-101, 2015.

[18] B. Hayes-Roth, R. Washington, D. Ash, R. Hewett, A. Collinot, A. Vina, and A. Seiver, "Guardian: A prototype intelligent agent for intensive-care monitoring," *Artificial Intelligence in Medicine*, vol. 4, no. 2, pp. 165-185, 1992.

[19] E. E. Millán-Rojas, J. N. Pérez-Castillo, and A. P. G. Torres, "Visualization of urban floodplains in the Amazon foothill using the geo-inspired model of natural vector multi-agents (AVNG)," *Revista Facultad de Ingeniería*, vol. 26, no. 45, pp. 173-186, 2017.

[20] M. Challenger, G. Kardas, and B. Tekinerdogan, "A systematic approach to evaluating domain-specific modeling language environments for multi-agent systems," *Software Quality Journal*, pp. 1-41, 2015.

[21] O. Dikenelli, R. C. Erdur, G. Kardas, Ö. Gümüs, I. Seylan, Ö. Gürcan, and E. E. Ekinci, "Developing multi agent systems on semantic web environment using seagent platform," in *International Workshop on Engineering Societies in the Agents World*, 2005, Springer Berlin Heidelberg, pp. 1-13.

[22] O. Dikenelli, Ö.Gümüs, A. M. Tiryaki, and G. Kardas, "Engineering a multi agent platform with dynamic semantic service discovery and invocation capability," in *Proc. German Conference on Multiagent System Technologies*, 2005, Springer Berlin Heidelberg, pp. 141-152.

[23] B. Hojat and M. Amamiya, "Ensuring the localization of responsibilities in the adaptive agent oriented software architecture (aaosa)," *Scientia Iranica*, vol. 8, no. 4, pp. 229-240, 2001.

[24] K. R. Malik, R. R. Mir, M. Farhan, T. Rafiq, and M. Aslam, "Student query trend assessment with semantical annotation and artificial intelligent multi-agents," *Eurasia Journal of Mathematics, Science and Technology Education*, vol. 13, no. 7, pp. 3893-3917, 2017.

[25] S. Wilk, M. Kezadri-Hamiaz, D. Rosu, C. Kuziemsky, W. Michalowski, D. Amyot, and M. Carrier, "Using semantic components to represent dynamics of an interdisciplinary healthcare team in a multi-agent decision support system," *Journal of Medical Systems*, vol. 40, no. 2, p. 42, 2016.

[26] K. P. Ferentinos, K. G. Arvanitis, D. Lambrou, A. Anastasiou, and N. Sigrimis, "A multi-agent system for integrated production in greenhouse hydroponics," in *Proc. International Conference on Sustainable Greenhouse Systems-Greensys*, 2004,vol. 691, pp. 381-388.

[27] S. H. Kasaei, S. M. Kasaei, and S. A. Kasaei, "Design and development a control and monitoring system for greenhouse conditions based-on multi agent system," *BRAIN. Broad Research in Artificial Intelligence and Neuroscience*, vol. 2, no. 4, pp. 28-35, 2011.

[28] J. E. Sagula and R. Teseyra, *MASICoG Multi-Agent System for the Intelligent Control of Greenhouses*, 2005.

[29] D. Stipanicev, M. Stula, and L. Bodrozic, "Multiagent based greenhouse telecontrol system as a tool for distance experimentation," in *Proc. the IEEE International Symposium on Industrial Electronics*, ISIE , 2005, vol. 4, pp. 1691-1696.

[30] D. Xu, and H. Li, "Intelligent greenhouse control-System based on Agent," in *Proc. International Conference on Intelligent Computation Technology and Automation (ICICTA*), 2008, vol. 1, pp. 390-394), IEEE.

[31] J. Ferber, "Multi-agent systems: An introduction to distributed artificial intelligence," *Reading: Addison-Wesley*, vol. 1, 1999.

[32] N. R. Jennings and M. Wooldridge, "Applications of intelligent agents," Agent Technology, pp. 3-28, Springer Berlin Heidelberg, 1998.

[33] M. Luck, P. McBurney, and C. Preist, "Agent technology: Enabling next generation computing (a roadmap for agent based computing)," AgentLink/University of Southampton, 2003.

[34] F. L. Bellifemine, G. Caire, and D. Greenwood, "Developing multi-agent systems with JADE," *John Wiley & Sons*, vol. 7, 2007.

[35] A. S. Rao, and M. P. Georgeff, "Modeling rational agents within a BDI-architecture," *KR*, vol. 91, pp. 473-484, 1991.

[36] H. Kuijs, C. Rosencrantz, and C. Reich, *A Context-aware, Intelligent and Flexible Ambient Assisted Living Platform Architecture*, Cloud Computing, 2015.

[37] Y. Labrou, T. Finin, and Y. Peng, "Agent communication languages: The current landscape," *IEEE Intelligent Systems*, vol. 14, no. 2, pp. 45-52, 1999.

[38] D. Gavalas, D. Greenwood, M. Ghanbari, and M. O'Mahony, "Advanced network monitoring applications based on mobile/intelligent agent technology," *Computer Communications*, vol. 23, no. 8, pp. 720-730, 2000.

[39] M. Baldoni, C. Baroglio, and F. Capuzzimati, "Programming JADE and Jason agents based on social relationships using a uniform approach", *Advances in Social Computing and Multiagent Systems*, pp. 167-184. Springer International Publishing, 2015.

[40] O. Boissier, R. H. Bordini, J. F. Hübner, A. Ricci, and A. Santi, "Multi-agent oriented programming with JaCaMo," *Science of Computer Programming*, vol. 78, no. 6, pp. 747-761, 2013.

[41] S. Ergün and T. Aydoğan, "JADE etmen çerçevesinde çok etmenli bir ders yönetim sisteminin sabro metodolojisi kullanilarak geliştirilmesi," *SDÜ Fen Bilimleri Enstitüsü Dergisi*, vol. 17, no. 3, 2013.

[42] S. Ergün and T. Aydoğan, "Kavşak sinyalizasyon sisteminin jack etmen geliştirme platformunun kullanilarak oluşturulmasi," *International Journal Of Informatics Technologies*, vol. 6, no. 1, pp. 8-16, 2013.

Tuncay Aydoğan received the PhD degree in 2005 from Sakarya University. He is currently an associated professor of Suleyman Demirel University. His research interests include smart systems, fuzzy logic, automatic control and ındustrial networks.

Serap Ergün is a research assistant at the Süleyman Demirel University. Her research interests include intelligent agents, artificial intelligence, game theory, network routing.