Journal of Universal Computer Science, vol. 15, no. 13 (2009), 2485-2505
submitted: 31/10/08, accepted: 13/6/09, appeared: 1/7/09 © *J.UCS*

A Multiagent System for Hierarchical Control and Monitoring

Vu Van Tan, Dae-Seung Yoo, Jun-Chol Shin, and Myeong-Jae Yi[1]
(School of Computer Engineering and Information Technology
University of Ulsan, Ulsan 680-749, Republic of Korea
{vvtan, ooseyds, sjc333, ymj}@mail.ulsan.ac.kr)

Abstract: This paper presents the architecture of a multiagent system based on new OPC Unified Architecture (UA) technology for hierarchical control and monitoring of a complex process control system. This architecture is proposed with utilization of the OPC technology, which contains both a continuous-event component and a discrete-event component by incorporating XML for the negotiation and cooperation with the multiagent system's environments. The practical applications of the proposed architecture are provided and the discussion demonstrates that the proposed architecture is both reliable and effective for applying to multiagent-based complex control system applications.
Key Words: Hierarchical control, monitoring, multiagent, OPC, process control, unified architecture, XML.
Category: C.2.4, D.2.10, D.2.11, D.2.12, H.1.0, H.4.1, H.4.3, K.1.0, L.3.1

1 Introduction

Agent-based system technology has attracted great interest in recent years because it represents a new paradigm for conceptualizing, designing, and implementing software systems. Currently, the vast majority of agent-based systems comprises a single agent. As the technology has matured in time and been used to address increasingly complex applications, the need for systems that consist of multiple agents in a peer-to-peer model has increased.

In addition to automation and information systems designed for industrial plant floors, they are more larger and complex in terms of architecture, platforms, and components including a lot of different components and different platforms such as control instrumentations, control softwares, and communication networks. However, multiagent technology could be helpful for process control, as has been pointed out by several researchers [Choinski et al. 2007a, McArthur and Davidson 2005, Davidsson and Wernstedt 2002a, Seilonen 2006, Rolle et al. 2005].

A complex process control system, i.e., a process automation system, can be characterized as a distributed and integrated monitoring, control, and coordination system with partially cyclic and event-based operations. Its control

[1] Corresponding author.

functions can be divided into continuous, sequential, and batch control. The role of continuous control makes a process control system different from other systems like discrete manufacturing systems. In addition to control functions, the complex process control system has other functions including performance monitoring, condition monitoring, abnormal situation handling and reporting. Complex process control systems are often distributed systems that consist of several controllers running interrelated control applications. The need for these systems is to integrate them and to coordinate their control operations. A complex process control system should utilize different hardware and software technologies to accomplish its tasks. The main challenge is the appropriate synthesis of flexible network-based computer and control systems. Flexibility is required to access and update the process data, events, operator and external expert decisions, and negotiations. The OPC Foundation recently defined new specifications for next generation for process control and monitoring systems running on various platforms, i.e., the OPC Unified Architecture (UA) specifications [The OPC Foundation 2007]. These specifications provide a paradigm for the design and implementation of control softwares.

Implementation of control tasks for complex process control systems is still challenging due to the complexity of related decision tasks and information systems. It is hard to integrate more advanced control strategies into real-world situations because of incompatibilities with the software systems already in place and difficulties in integrating the process control system. Although several approaches to implement multiagent-based complex process control systems have been successfully proposed and developed in recent years, these are not yet capable of providing an ultimate solution and have limited functionality, e.g., they do not incorporate monitoring operations.

The study of this paper aims at proposing and developing a multiagent system for hierarchical control and monitoring of complex process control systems using the new OPC UA specifications. This system allows us to access and update the process data, events, and operator and external expert decisions, among others. The proposed system is designed and developed with combining databases, resulting in a flexible method that can be used in various applications. It also addresses the fragmented state of the current research in terms of multiagent applications of complex process control systems, and incorporates both the control and information access operations in process automation.

This paper is organized as follows: The next section shows an overview of the existing multiagent frameworks that have been proposed in recent years and provides the weaknesses under consideration of these frameworks. Section 3 proposes an OPC-based architecture for a multiagent system for hierarchical control and monitoring of process control systems that addresses the problems mentioned in Section 2. The architecture, levels of agents, agent model, and modules of implementation of the proposed system are also described in detail. The design of database structure is introduced in Section 4. The practical applications to demonstrate that the proposed architecture is reliable and feasible for applying to various applications, and the discussion of the proposed system are presented in Section 5. Finally, Section 6 will mark some major conclusions and future directions.

2 Related Work

Agents have been applied in several domains [Sycara 1998]. In general, the applications of agents can be divided into two main categories, namely *distributed systems* and *personal software assistants* [Wooldridge 2002]. A multiagent system consists of multiple agents linked by some type of cooperation mechanism. Each agent can implement tasks independently. Through cooperation and interaction among agents, a multiagent system can accomplish and optimize complex tasks. Process control systems can be classified into three types on the basis of their control architectures: *centralized control*, *hierarchical control*, and *decentralized control*. However, centralized control is unreliable and inflexible because only one supervision unit is used. Control frameworks can alternatively be classified into *hierarchical*, *heterarchical*, and *hybrid* control frameworks [Heragu et al. 2002]. The hierarchical approach assumes that there is a hierarchy and a master-slave relationship between higher and lower levels of controls. The hierarchy is introduced to handle the complexity of a system [Leduc et al. 2006, Dong et al. 2006]. The heterarchical approach focuses on interactions between unit controllers to allow system flexibility, while the relationship between higher- and lower-level controllers is ignored. The hybrid approach has features of both hierarchical and heterarchical approaches. It allows direct interactions among the lower-level controllers as well as between higher and lower levels [Nahm and Ishikawa 2005, Maturana 1997, Voos 2000]. In recent years, a large number of multiagent systems have been successfully proposed, developed, and applied to real-world environments. Multiagent technology has also been applied to improve nonconventional biotechnological process control in a pilot-plant effectively [Nocon et al. 2004, Choinski et al. 2007b].

Jennings [Jennings 1995] has introduced a multiagent system development project to develop and deploy multiagent technology in several industrial domains. The most significant domain is the power distribution system. Agents used in this system have two main components: 1) a *domain* component, which realizes the domain-specific functionality of the agent and 2) a *wrapper* component, which provides the agent with the functionality to enable the system to plan its actions and to present its actions and communicate with other agents.

Choinski et al. [Choinski et al. 2007a] described a hierarchical, control, and information system with the capacity to update a self-organizing database in real-time and to negotiate control events and decisions. This system allows decomposition of complex biotechnological object with control hierarchical system and analysis of this system for time-driven and event-driven parts. Communication is ensured by the presence of both a client-server and producer-consumer protocol. Three types of agents varying in their degrees of cooperation are produced by the data model of this system: *competitive agents*, *collaborative agents*, and *hostile agents*. However, the monitoring operations used for monitoring and control systems were not incorporated in this system.

Davidsson and Wernstedt [Davidsson and Wernstedt 2002a] argued for the appropriateness of using software agents to monitor and control bioprocesses. Bioprocesses are getting more complex and often achieve both economical and environmental constraints. They lead to modeling and control problems of increasing complexity when trying to build systems that operate robustly over a wide range of conditions. Different approaches to the design of individual agents as well as complete multiagent systems are discussed.

Software agents are especially suited to the design and implementation of systems that are partitioned into smaller operating entities [Dong et al. 2006, Davidsson and Wernstedt 2002a, Voos 2000, Guo et al. 2006]. An industrial process is usually equipped with a number of database handlers, control system or expert systems that are not agent-ready. Thus wrapper agents are used to provide an approach for systems that already used.

Thomas et al. [Thomas et al. 2001] proposed and developed new prototype tools under `agent command` to enable the post-simulation monitoring and analysis of the object-agent message traffic. Nevertheless, these tools do not enable real-time command and analysis of distributed agent systems in terms of monitoring operations.

Fregene et al. [Fregene et al. 2005] developed a system- and control-oriented intelligent agent framework that functioned as a hybrid intelligent control agent. This framework was used to control of teams of unmanned air and ground vehicles. However, the issue of communication delays and the effect of these delays on the transmission and reception of supervisory and coordination signals were not discussed. Furthermore, no monitoring operations were included in this agent framework.

Nahm and Ishikawa [Nahm and Ishikawa 2005] proposed a hybrid agent architecture that enables agents to exhibit hybrid behavior and hybrid interaction for building lightweight, dynamic, and large scale multiagent systems. In addition, the collaborative product development environment was also illustrated by successful implementation of a multidisciplinary design problem over the Internet. This architecture can be applied in many intelligent manufacturing systems for virtual enterprise environments.

Dong et al. [Dong et al. 2006] proposed an architecture for a multiagent system for industrial process control. This architecture consists of three layers: a *cooperation layer*, a *planning layer*, and a *control layer*. The control layer is a reactive layer with real-time characteristics. The cooperation layer is defined as a self-existent layer and is open and flexible, which is helpful to integrate the multiagent system with the existing systems. The planning layer has main processes such as *plan*, *negotiate consistency*, *interfaces*, *information services*, etc. However, although the components of the three layers were described, no validation of the performance was provided.

Najid et al. [Najid et al. 2002] used a multiagent approach with distributed hierarchical control for the flexible manufacturing cell. There are two layers in their model: a *coordinator layer* and a *lower layer*. Decision can be taken either in an individual or a cooperative way. Their approach is composed of a set of cognitive and reactive agents. The control decisions are made according to the actual state of the manufacturing system. As a result the levels of control are limited and this architecture did not incorporate monitoring operations.

Damba and Watanabe [Damba and Watanabe 2007] introduced a multiagent system with a hierarchical representation designed for efficient control. Hierarchical representation is ensured by a problem decomposition structure. This system is based on agent modeling features such as *individual* and *social criteria*. It consists of multiple agents at different levels of the hierarchy of a control system. At each level of the hierarchy, the task is reduced to a small number of actions at the next lower level to improve the computational cost of finding the correct way to arrange the actions. However, this multiagent system was only used as a game simulation to verify the suggested solution, and was not applied to process control systems.

Kendall and Malkoun [Kendall and Malkoun 1997] presented a design for a multiagent systems using *object-oriented design patterns*. This approach addresses agent concurrency, virtual migration, collaboration, and reasoning based on the role of design patterns. On the other hand, Xue et al. [Xue et al. 2006] proposed a customized methodology to develop multiagent systems in order to realize engineering changes in agent-oriented software engineering. Their method can cover the whole development life-cycle from agent-oriented analysis to software implementation. Developers then use it to combine different meta models originating from various methods to produce the optimal suited development process for a particular application domain. However, it is difficult to alter these design patterns or the mentioned methodology for applications in complex process control systems [Mönch and Stehli 2006].

Although the existing approaches have focused on the control functions of process control systems, they have been limited in attention to other functions, e.g., monitoring operations [Seilonen et al. 2005]. Moreover, research on the application of multiagent systems in process control and monitoring has been less extensive than in discrete manufacturing. One reason for this may be that it is is challenging for the current agent technology to meet the real-time requirements of process control and monitoring applications. The second reason is that process control tasks are usually characterized by complex interrelationships between various controlled variables. This makes us difficult to find problem decompositions that are suitable for agents [Bussmann et al. 2001, Seilonen 2006]. A third reason is that the suitability and usefulness of multiagent systems in process control are not more evident than in discrete manufacturing.

In this paper, a novel architecture for a multiagent system for hierarchical control and monitoring of complex processes is proposed using the new OPC UA technology that provides a powerful communication technology to support both real-time and non-real-time applications. The architecture, levels of agents, modules of implementation, and database structure are described in the next sections. The use of the OPC technology makes the system flexible and open, which allows the development of various agent-based complex process control system applications for hierarchical control and monitoring. This system architecture conforms closely to the specifications of the FIPA standards[2] for agent technology. The practical applications of the proposed architecture are discussed, and comparisons with existing multiagent systems are also provided. They demonstrate that the proposed architecture can be used to deploy several agent applications in current industry.

3 Design of the Multiagent-based Process Control System

This section will introduce the design of a multiagent-based control system, including its architecture, levels of agents, agent model, modules of implementation, and functionality. The design of the database structure to which agents require access is detailed in Section 4.

3.1 Architectural System Design

In general, the type of an agent depends on the way of the state changes and on the knowledge of the agent. The agents are specified on a design level in an iterative process in addition of knowledge as needed. The hierarchical control system model of the agents is shown in Figure 1. There are three kinds of agents as the followings [Choinski et al. 2007a]:

- **Control agent**. This agent uses a trajectory for state transition and provides the basic control algorithm for the process. Generally, all the closed-loop and open-loop control algorithms are implemented by this agent. The

[2] FIPA specifications are a collection of standards intended to promote the interoperation of heterogeneous agents and the services that they can represent. Details about the specifications can be found at `http://www.fipa.org`

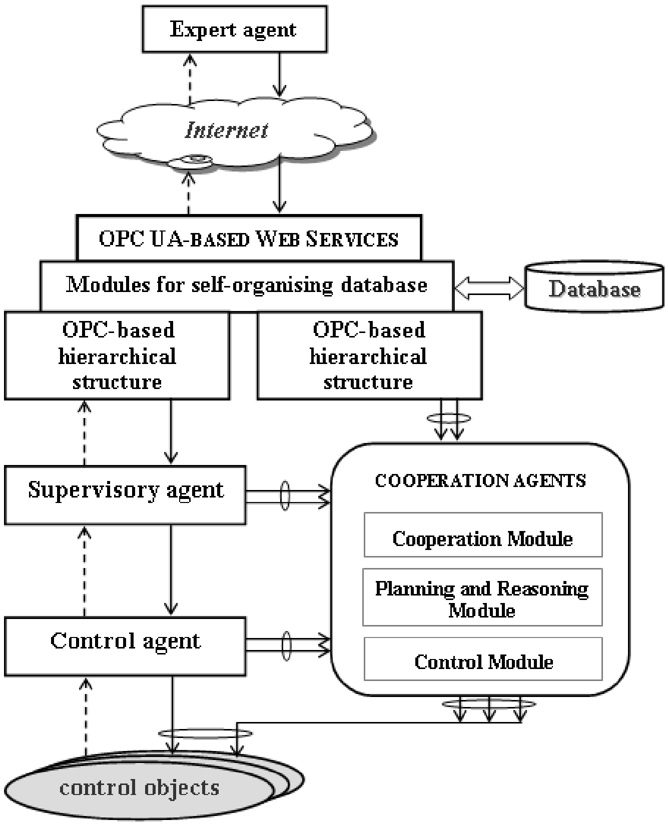

Figure 1: The proposed architecture of a multiagent system and its agent levels.

agent's knowledge is based on a control algorithm and its effectors are considered to be *control actions*.

– **Supervisory agent**. This agent is responsible for general supervision of the process performance. It provides supervisory agent compensation against process fluctuations by executing a special sequence of operations. The knowledge of this agent depends on phenomenological models of the object and operator experience, and the specific effector is considered to be a *supervisory compensator*.

– **Expert agent**. This agent provides remote expert knowledge for off-nominal situations that cannot be dealt with by the supervisory or control agent. Its effector is considered to be an *expert compensator*.

The agents in the multiagent systems not only communicate with the users and objects, but also communicate and incorporate with other agents. To solve these problems, direct control interaction can be used to allow agents to cooperate with control system functions. Each agent can initiate a state transition, but not every agent will actually enforce this agent without cooperation from the other agents. In addition to industrial systems, distributed control systems (DCSs) are normally divided into two types by behaviors such as time-driven and event-driven, respectively.

An important issue in general multiagent systems is that agents should be able to cooperate and communicate with others. Agents should share their knowledge in order to maximize the benefits of communication in critical system controls. For example, they may start negotiating with each other and try to resolve the parameters of controllers using phenomenological models and on-line measurement identification. In the proposed cooperation model provided by the different levels of abstraction, the mappings of real objects of the controlled objects can be represented in the `address space` as a hierarchical structure based on the OPC UA technology. Moreover, the supervisory agent is equipped with models, methods, and algorithms to additional indirect controls. Off-line measurements, knowledge of the expert agent user, and additional algorithms with supervisory agent methodology are virtual levels of the database. They make us relatively easy to identify, modify, and reuse components for several applications. By combining information from field device measurements, operational state classification, condition monitoring, and process models, the monitoring operations based on distributed searches, processing, and monitoring of information are guaranteed. Therefore, the proposed system has several differences with the multiagent system developed by Choinski et al. [Choinski et al. 2007a].

The suggested architecture of the multiagent system based on the OPC UA technology is illustrated in Figure 1. This architecture model indicates how the hierarchical control levels are structured and connected, and provides more detailed information on the guarantees of the two types of DCS behaviors that the cooperation model has to incorporate. As Figure 1 shows, the data or events from the control objects and parameters are stored in a relational database after calculating the corrected values based on the control algorithms.

3.2 Control and Monitoring Operations for the Decision-Making Processes

The process agents perform supervisory control operations either in a sequential or iterative fashion. These agents process state changes or batch control operations and the later to tuning of continuous control. The task of the agent is first to plan a shared sequence of control actions and then to execute this sequence. These agents calculate optimal values for the supervisory control variables. Therefore, the decision-making processes of the agents are different depending on the role of the iterative refinement of the supervisory control variables. The input to the local decision-making of a coordinator agent is composed of the global control objective and the current values of process measurements. The input to the local decision-making of participant agents consists of the goal received from the coordinator and the current values of process measurements. The results from the decision-making tasks of the agents are new control variable values.

The monitoring agents perform monitoring operations in a distributed manner. Their operations combine information from the field-device measurements, operational state classifications, simulations, condition monitoring, and process models [Seilonen et al. 2004]. The operations of the monitoring agents are based on distributed searches, processing, and monitoring of information. Information searches are decomposed using an understanding of the physical structure of the monitored process, its present state, various diagnostics reports, and the ability of different information providers. The monitoring operation is an active watchdog that is needed to successfully supervise changes in some part of the process related data. A reasoning operation is used to produce derived information on the process performance based on the basic monitoring data and its modeled relationships.

Agents can share their knowledge to maximize the benefits of communication and try to resolve process measurements values and the results of local decision-making processes to achieve new values of control variables using their *self-learning capability*. The negotiation process may be carried out using vertical and horizontal cooperation channels. On the one hand, a supervisory agent can assign subgoals of their own goals to their subordinates. On the other hand, peer agents can negotiate with their peers in order to handle interrelations between their goals. The negotiations may be chained, i.e., one negotiation may be started by another one.

Both types of agents described above communicate according to the FIPA specifications. The process agents form a hierarchy based on authority relations. The leaf agents supervise some parts of the controlled process and its related automation system. In addition, the monitoring agents cooperate with the process agents to perform advanced monitoring tasks.

3.3 Agent Model for the Process Control and Monitoring

The agent model of an automation agent[3] specifies the internal modules of an agent and its operations. However, automation agents need to conform to

[3] This term is used to indicate either a process agent or a monitoring agent.

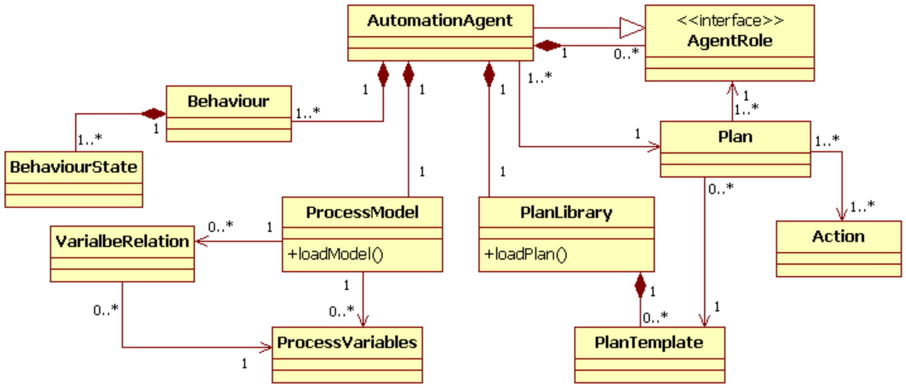

Figure 2: UML class diagram for describing components of the automation agent.

the agent models of some FIPA-compliant generic agent platform. In general, an agent consists mainly of a set of behaviors and actions that define the agent's reaction to different circumstances such as incoming messages or events [Seilonen 2006]. The agent's behaviors, which form the basis for reactivity and pro-activity, are used to implement agent interactions. An automation agent will use its actions to fulfill its goals. The automation agent is composed of modules that can be classified into operational and modeling modules and runtime data structures. This is illustrated in Figure 2.

The process model describes the knowledge of an automation agent on the controlled process. Automation agents can have knowledge about the existence of process variables, measured values, and relation between variables. A control variable may be controlled by one agent only, whereas the measurements can be shared among several agents. The plan library contains plans that an automation agent can use when creating runtime plans. Each automation agent is configured during agent application development with a set of plans that it needs to be able to plan its control operations. The runtime data structure of an automation agent includes goals, runtime plans, and contracts. The role of data structures is to store information on which the agent's goal is currently trying to fulfill.

A multiagent system for hierarchical control and monitoring of a complex process control system can be modeled using different goals depending on the different capabilities of the system. This allows the reconfiguration during runtime. Furthermore, adaptability and reconfigurability can be built into the system using agent role-based implementation to enhance or expand the functionalities of an agent-based complex process control system in order to meet the changes in technology.

3.4 Modular Design

The modules for the implementation of the proposed hierarchical control and monitoring system for multiagent-based complex process control are then introduced. These components can be described as follows:

1. The modules of a self-organizing database are used for processing XML data between control objects and web services and then data or events are stored in additional tables in the relational database. Moreover, these modules also have methods to connect the proposed system to the specific relational databases, e.g., Microsoft SQL Server, My SQL, etc.

2. The modules of interfaces and control algorithms are implemented as web services to provide both interfaces for the end-users and control algorithms to process data from the control objects.

3. Modules based on the OPC UA specifications to provide mechanisms such as session management, control object management, subscription management, and methods for *reading/writing* data *from/to* control objects (hardware I/O devices or endpoints) are designed and developed.

4. The model used to facilitate cooperation between multiple agents in the hierarchical control system is shown in Figure 1. The modules that need to corporate include the *cooperation module*, *planning and reasoning module*, and the *control module*. As the cooperation model is provided by two different levels of abstraction, the real object level of the controlled objects is represented as a hierarchical structure complying with the OPC UA model. The structure of the OPC UA data model provided by the OPC UA specifications can be used to decide which part of DCSs is available for the multiagent system. The virtual object level is based on database information according to the models and knowledge of the expert agent and the additional algorithms with the methodology of a supervisory agent.

In brief, the implemented modules allow the application developers to build a multiagent system for the process control and monitoring using the composition and inheritance of a set of restricted parts. They are a considerable effort in the development of the system design. In addition, by considering a data model of a hierarchical control system, which is based on the OPC UA data model, the data model stored at the OPC server can be expended for particular OPC clients as needed. Therefore, a general data model should be proposed for application developers.

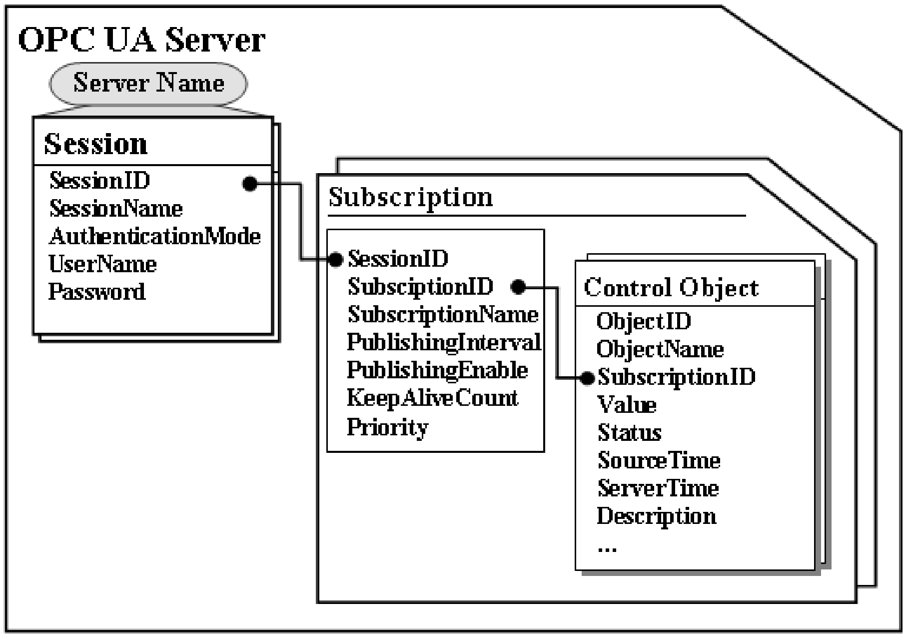

Figure 3: The data structure of the OPC UA server model used in the proposed multiagent system for hierarchical control and monitoring.

4 Design of the Database Structure

Because agents require to access a database to be able to participate in a system where information is exchanged, the connection between the proposed system and relational database standards must be guaranteed. The general model for the database structure of the proposed system, shown in Figure 3, has a flexible architecture that allows it to be used for several complex process control system applications. This feature makes the database design easy to modify and improve if the service developers wish to implement the specific applications. In general, the information generated by the OPC server-client model is stored in the additional tables created by the specific database that connects to the proposed system.

The design of the additional tables and their relationship to provide details about the database structure are shown in Figure 4. This database structure has a common design based on the OPC UA technology to store data, events, and parameters from the control objects. However, several tables can be added to this database structure depending on the requirements or goals of a specific application, ensuring that various real applications of process control systems

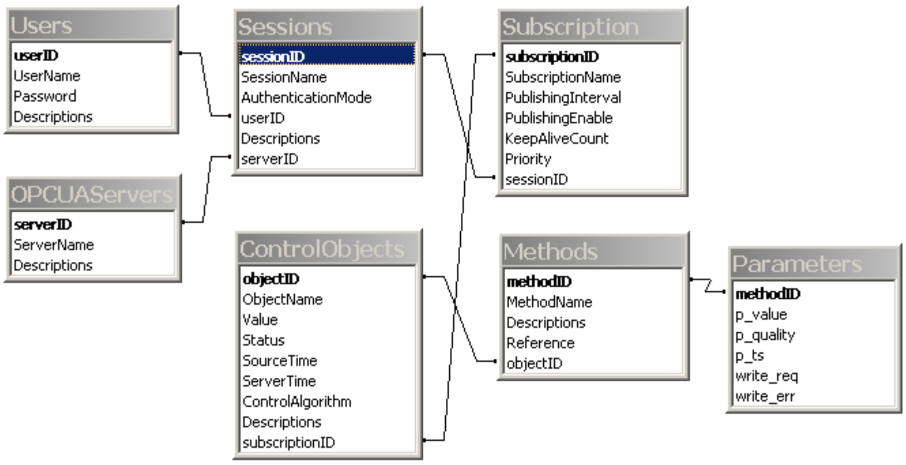

Figure 4: The tables and relationships among the additional tables in the relational database for the proposed architecture.

can be satisfied.

A brief description of the additional tables designed to store the data, events, and other information follows. The *Users* table is used to save a user's information such as the *user identifier* and *password* when the end-user logs onto the system. To store the session parameters such as the *session identifer*, *session name*, *authentication mode*, etc., the *Sessions* table in which each session may often consist of more subscriptions is used. The *Subscriptions* table contains several fields including *subscription identifier*, *subscription name*, *publishing interval*, *publishing enable*, and so forth for each subscription established by a user. A group of *control objects* added by users is represented in a particular subscription. Each control object normally contains some important information including *value*, *status code*, *source time*, *server time* (i.e., `timestamp`), etc. A *ControlObjects* table is then created. This, in turn, enables a hierarchical structure and data coherence to be maintained. In a normal case one *subscription* is composed of a large number of *control objects*. In addition, the two tables of *Methods* and *Parameters* are designed to store information on the control algorithm and its parameters, respectively. In summary, additional tables are dependent on the data structure of the OPC server and the control algorithms. They ensure that the proposed system has flexibility.

When the expert chooses interesting items (i.e., objects for reading and writing), a control algorithm is defined to operate on items that where defined in the OPC server for reading and calculating values that are overwritten in items defined for writing. Items selected by the expert agent constitute the expert's configuration and should be saved in the database. The saved configuration will be automatically restored when the browser-based client starts. When the expert agent processes the defined configuration, it can run the `read/write` mode in the application. This phase is repeated until the expert agent algorithm is stopped. If the expert agent decides that a single `write` operation is needed for a selected object, but a supplement of the algorithm is not planned, it will take advantage of an individual write.

The appropriate mechanisms for the execution of the proposed system are included the following steps:

1. Downloading the current values from the additional tables designed in the database and then storing these values as control variables.
2. Executing the control algorithms installed to calculate the new values that will be stored in the database, i.e., data or events and parameters are updated to the additional tables automatically.
3. Writing the changed values from the execution of the control algorithms to the additional tables and the control objects represented at the `address space` that is configured by the configuration users or administrators.
4. Sending the calculated values created by the control algorithms to the OPC server-client model. These values obviously depend on the specific control algorithms that are implemented.

The proposed system requires that the expert agent must login to the system and download control objects, i.e., OPC items in the `address space`, which are available in the database. These objects directly reflect measurement data, off-line laboratory data, control algorithm parameters, etc. The UML[4] sequence diagram to describe the interactions of the proposed system based on the above four steps is shown in Figure 5.

5 Practical Applications and Discussion

This section considers the use of the proposed system through practical application as case studies. Moreover, the discussion of the proposed system is presented. A qualitative comparison between the proposed system and existing approaches is made. This comparison indicates the proposed system's openness, flexibility, robustness, and reusability.

[4] Unified Modeling Language

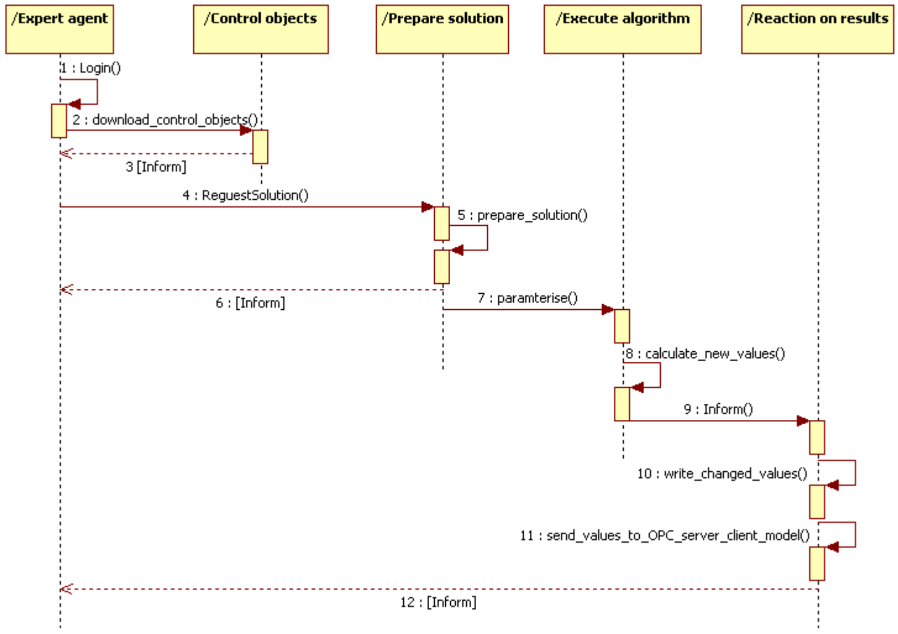

Figure 5: Interactions of the proposed system according to the four mentioned steps to calculate new values for the process control system.

5.1 Applications of the Proposed Architecture

In contrast to more traditional process control strategies that are based on mathematical process models, recent research has been focused on new techniques typically borrowed from the fields of artificial intelligence. Agent technology, which can be applied to the process monitoring and control, is a suitable integration technique and, in addition, makes it possible for control to be distributed rather than centralized [Davidsson and Wernstedt 2002a]. The basic motivation for using agents in monitoring and control applications is to distribute tasks both physically and logically. Therefore, distributed solutions are often more easier to understand and develop than centralized solutions. They are particularly useful in situations where controllers are distributed physically.

Several multiagent-based approaches were successfully proposed, developed, and applied to manufacturing and process control systems today. Additionally, the OPC UA specifications for the design of these systems were proposed by the OPC Foundation[5] that comprises leading manufacturers and solution providers in factory and process automation as well as enterprise solutions. By

[5] `http://opcfoundation.org`

systematically reviewing a number of approaches and frameworks that have been focused on agent-based process control systems, a novel architecture for a multiagent system for hierarchical control and monitoring of a process control system is proposed, which provides a unified mechanism to the domain of complex process control systems. The proposed architecture was presented including interfaces, levels of agents, modules of implementation, and a common database structure as foundations to apply to agents-based process control systems for hierarchical control. To illustrate the ability of the proposed architecture, the use of the proposed system is indicated through some related applications that have been successfully developed by incorporating other technologies [Heragu et al. 2002, Oh et al. 2007, Guo et al. 2006].

For example, an application for pressure control of recycled gas developed by Guo et al. [Guo et al. 2006] can be solved by the proposed architecture. In this example, recycled gas is the main fuel used to provide energy to produce coke in a coke plant. The stability of the pressure of the recycled gas directly influences the stability of inner temperature of the coke oven, which in turn affects the quality of the coke and the quality of gas. However, even through pressure control of the recycled gas is critical, it can be disturbed by various factors. Normally, pressure control systems emphasizes particularly on different control performance in different circumstances. Multiagent-based pressure control of the recycled gas can be achieved using a hybrid control mode that combines hierarchical structure and decentralized control units. The second example deals with the monitoring and control of district heating systems [Davidsson and Wernstedt 2002b]. The basic idea behind district heating is to use cheap local heat production plants to produce hot water. This water is then distributed through pipes to the customers who use it as a source of heated tap water and/or to heat the building. At the customer side, substations are used to exchange heat from the primary flow of the distribution pipes to the secondary flow of the building. The main problem encountered by the control engineer is to decide the correct amount of heat to produce at each point in time. Another problem is that if there is a shortage of heat in the system, the control engineer needs to make sure that consumers receive appropriate amounts of heat according to the circumstances. To resolve these problems the proposed architecture has two functions: 1) serves as a decision support system for the control engineer by providing information on the current state and predictions of future states and 2) has a fully automated control mechanism for dealing with heat shortages. Heat prediction is based on the average consumption during the corresponding time period. It continuously monitors the heat consumption and sends a report to the redistribution agent. Two other examples that can be solved by the proposed architecture are described by Davidsson and Wernstedt [Davidsson and Wernstedt 2002a].

5.2 Comparison with Existing Approaches

An OPC-based multiagent system for hierarchical control and monitoring of complex process control systems was proposed. The use of agent technologies and the OPC technology makes the proposed system flexible, robust, and open. A *unified and flexible* architecture for the implementation of an agent-based process control system for storing data, event, and control parameters was proposed, and the design of the common database structure was described. This database contains additional tables that can be modified if the system is applied to different applications of complex process control systems. As a result, the proposed system is flexible and the cooperation capability among agents allows the integration of hierarchical control and decentralized control. The proposed system allows us to access and update process data, events, and operator and external expert decision based on the database connected to the system.

In terms of control architectures, centralized control seems to be unreliable and inflexible because information about all controllers is supervised by one supervisory. Therefore, the hierarchical control, in which the functions of supervision are divided vertically, was proposed to deal with complexity. However, horizontal division of control functions is not included. It is obvious that the architecture which combines a hierarchical structure and decentralized control units is reasonable and flexible. The comparison of the proposed architecture with other approaches is difficult due to the conceptual nature, different architectures, and the wide range of production environments. A qualitative comparison of the proposed architecture and existing frameworks could be made. Thus the structural characteristics of the proposed system are used to compare with others. The proposed architecture of a multiagent system for process control systems with hierarchical control issue has the following advantages:

1. By using on new features of the OPC UA technology, the control address space can collect current data and events that are truly related to each others. The concept of the proposed architecture introduces a new level of design and implementation that provides an easier conceptualization of the control agent domain. This architecture, of course, enables the implementation of more complex control strategies where the control is distributed. It can also be used to develop control strategies where large amounts of distributed data are collected.

2. While other proposals have been designed for entire manufacturing systems and therefore include product development, product planning, process planning generation, and scheduling and control, e.g., `PROSA` [Brussel et al. 1998], the proposed system was designed primarily for process planning, scheduling and control, and production control, which are achieved by integrating the

hierarchical control and decentralized control, making the proposed system more flexible than existing multiagent systems.

3. The proposed system has a self-organizing database that is used to ensure that data or events are stored automatically in a relational database. The database structure is based on the OPC UA data model to ensure flexibility of the hierarchical model. This database structure is open to allow it to be accessed by application developers when designing and implementing novel agent-based process control system applications.

4. In contrast to several existing approaches that focus on the domain of manufacturing systems [Leduc et al. 2006, Maturana 1997, Heragu et al. 2002], the domain of the proposed system is that the hierarchical control and monitoring of complex process control systems are covered including the guarantees of updating the process data, events, operator and external expert decisions, and negotiations.

5. The proposed system was designed and developed in a systematic way to reach an ultimate solution. This system not only focuses on the control functions of complex process control systems, but also has attention to other functions, e.g., monitoring operations. The modules developed within the proposed system were designed for the purpose of developing an agent-based complex process control system with new roles for agents.

6. The proposed architecture can be applied to various process control and monitoring systems, in contrast to most existing approaches that focus only on the developments of specific applications [Wernstedt and Davidsson 2002, Davidsson and Wernstedt 2002a, Oh et al. 2007, Nahm and Ishikawa 2005, Guo et al. 2006, Choinski et al. 2007a].

The proposed architecture, which combines hierarchical structure with decentralized and hierarchical control units, is flexible, open, and easy to understand and implement for developers and programmers alike. It provides a mechanism, a design foundation, and a common framework for application developers to develop multiagent systems of OPC-based complex process control systems with a self-organizing database.

6 Conclusions and Future Directions

In this paper, the architecture of a multiagent system for hierarchical control and monitoring of complex process control systems has introduced. A generic architecture was suggested for the implementation of multiagent-based process control systems, allowing for a flexible integration of different control strategies.

This architecture permits us to access and update the process data, events, and operator and external expert decisions. Details about architecture, control and monitoring operations for the decision-making processes, agent model for the process control and monitoring, modules for the implementation of the proposed system, and the database structure are presented.

The proposed multiagent-system architecture can be applied in several multiagent system domains for hierarchical control and monitoring of complex process control systems as aforementioned. Depending on the purpose of a specific application of complex process control systems, the proposed system can be changed for adapting its requirements or goals. In addition, the applications of the proposed architecture to implement an agent-based complex process control system were highlighted through practical applications as case studies. The proposed architecture for the agent-based process control system is very flexible, robust, open, and also supports real-time and non-real-time constraints. The proposed multiagent system provides design solutions and a common framework for application developers when implementing various agent applications.

The future work is trying to use the proposed architecture for the implementation of an agent-based process control approach for flexible process control systems. The learning behavior of an agent based on its control algorithm while taking in the view of the attributes of objects will be also investigated.

Acknowledgements

The authors would like to thank the Korean Ministry of Knowledge Economy, Ulsan Metropolitan City, University of Ulsan, and the Network-based Automation Research Center (NARC), which partly supported this research. The authors also would like to thank the anonymous reviewers for their carefully reading and commenting this paper.

References

[Brussel et al. 1998] Brussel, H.V., Wyns, J., Valckenaers, P., Bongaerts, L., and Peeters, P.: "Reference Architecture for Holonic Manufacturing Systems: PROSA"; Computers in Industry, 37, 3 (1998), 255-274.

[Bussmann et al. 2001] Bussmann, S., Jennings, N.R., and Wooldridge, M.: "On the Identification of Agents in the Design of Production Control Systems"; Proceedings of the 1st International Workshop on Agent-Oriented Software Engineering, Lect. Notes in Comp. Sci. 1957, Springer, Berlin (2001), 141-162.

[Choinski et al. 2007a] Choinski, D., Nocon, W., and Metzger, M.: "Multi-Agent System for Hierarchical Control with Self-organising Database"; Proceedings of the 1st KES Symposium on Agent and Multi-Agent Systems-Technologies and Applications. Lect. Notes in Artificial Intelligence 4496, Springer, Berlin (2007), 655-664.

[Choinski et al. 2007b] Choinski, D., Metzger, M., Nocon, W., and Polakow, G: "Cooperative Validation in Distributed Control Systems Design"; Proceedings of the 4th International Conference on Cooperative Design, Visualization, and Engineering, Lect. Notes in Comp. Sci. 4674, Springer, Berlin (2007), 280-289.

[Damba and Watanabe 2007] Damba, A. and Watanabe, S.: "Hierarchical Control in a Multiagent System"; Proceedings of the 2nd International Conference on Innovative Computing, Information and Control, IEEE Press (2007), p. 111.

[Davidsson and Wernstedt 2002a] Davidsson, P. and Wernstedt, F.: "Software Agents for Bioprocess Monitoring and Control"; Journal of Chemical Technology and Biotechnology, 77, 7 (2002), 761-766.

[Davidsson and Wernstedt 2002b] Davidsson, P. and Wernstedt, F.: "A Multi-Agent System Architecture for Coordination of Just-in-time Production and Distribution"; Proceedings of the 17th ACM Symposium on Applied Computing, ACM, New York (2002), 294-300.

[Dong et al. 2006] Dong, J., Yin, Y., and Peng, K.: "A Hierarchical Control System Based on Agent Technology"; Proceedings of the IEEE International Symposium on Intelligent Control, IEEE Press (2006), 2731-2736.

[Fregene et al. 2005] Fregene, K., Kennedy, D.C., and Wang, D.W.L.: "Towards a Systems- and Control-Oriented Agent Framework"; IEEE Transactions on System, Manufacturing, and Cyberntics - Part B: Cybernetics, 35, 5 (2005), 999-1012.

[Guo et al. 2006] Guo, Y., Cheng, J., Gong, D., and Zhang, J.: "A Novel Multi-agent Based Complex Process Control System and Its Application"; Proceedings of the 2006 International Conference on Intelligent Computing, Lect. Notes in Control and Information Sci. (LNCIS) 344, Springer, Berlin (2006), 319-330.

[Heragu et al. 2002] Heragu, S.S, Graves, R.J., Kim, B.I., and Onge, A.S.: "Intelligent Agent Based Framework for Manufacturing Systems Control"; IEEE Transactions on Systems, Man, and Cybernetics - Part A, 32, 5 (2002), 560-573.

[Jennings 1995] Jennings, N.R.: "Controlling Cooperative Problem Solving in Industrial Multiagent System Using Joint Intentions"; Artificial Intelligence, 75, 2 (1995), 195-240.

[Kendall and Malkoun 1997] Kendall, E.A. and Malkoun, M.T.: "Design Patterns for the Development of Multiagent Systems"; Proceedings of the Second Australian Workshop on Distributed Artificial Intelligence: Multi-Agent Systems: Methodologies and Applications, Lect. Notes in Comp. Sci. 1286, Springer, Berlin (1997), 17-31.

[Leduc et al. 2006] Leduc, R.J, Lawford, M., and Dai, P.: "Hierarchical Interface-Based Supervisory Control of a Flexible Manufacturing System"; IEEE Transactions on Control Systems Technology, 14, 4 (2006), 654-668.

[Maturana 1997] Maturana, F.: "Multi-agent Architectures for Concurrent Design and Manufacturing"; Proceedings of the IASTED International Conference on Artificial Intelligence and Soft Computing, ACTA Press (1997), 355-359.

[McArthur and Davidson 2005] McArthur, S.D.J. and Davidson, E.M.: "Multi-Agent Systems for Diagnostics and Condition Monitoring Applications"; Proceedings of the 13th International Conference on Intelligent System Application to Power Systems, IEEE Press (2005), 201-206.

[Mönch and Stehli 2006] Mönch, L. and Stehli, M.: ManufAg: "A Multi-agent-System Framework for Production Control of Complex Manufacturing Systems"; Information Systems and e-Business Management, 4, 2 (2006), 159-185.

[Nahm and Ishikawa 2005] Nahm, Y.E. and Ishikawa, H.: "A Hybrid Multi-Agent System Architecture for Enterprise Integration using Computer Networks"; Robotics and Computer-Integrated Manufacturing, 21, 3 (2005), 217-234.

[Najid et al. 2002] Najid, N.M., Kouiss, K., and Derriche, O.: "Agent based Approach for a Real-time Shop Floor Control"; Proceedings of the 2002 IEEE International Conference on Systems, Man and Cybernetics, vol. 4, IEEE Press (2002), 6-9.

[Nocon et al. 2004] Nocon, W., Choinski, D., and Metzger, M.: "Web-based Control and Monitoring of the Experimental Pilot Plant Installations"; Proceedings of the IFAC Workshop on Programmable Devices and Systems, 2004, 94-99.

[Oh et al. 2007] Oh, S., Schenato, L., Chen, P., and Sastry, S.: "Tracking and Coordination of Multiple Agents Using Sensor Networks: System Design, Algorithms and

Experiments"; Proceedings of the IEEE, 95, 1 (2007), 234-254.

[Rolle et al. 2005] Rolle, M., Novak, O., Kubalik, J., and Pechoucek, M.: "Alarm Root Cause Detection System"; Proceedings of the 6th IFIP International Conference on Information Technology for Balanced Automation Systems in Manufacturing and Services, vol. 159, Springer, Berlin (2005), 109-116.

[Seilonen et al. 2004] Seilonen, I., Pirttioja, T., Pakonen, A., Appelqvist, P., Halme, A., and Koskinen, K.: "Modelling Cooperative Control in Process Automation with Multi-agent Systems"; Proceedings of the 2nd IEEE International Conference on Industrial Informatics, IEEE Press (2004), 260-265.

[Seilonen et al. 2005] Seilonen, I., Pirttioja, T., Pakonen, A., Appelqvist, P., Halme, A., and Koskinen, K.: "Information Access and Control Operations in Multi-agent System Based Process Automation"; Proceedings of the 2nd International Conference on Applications of Holonic and Multi-Agent Systems. Lect. Notes in Comp. Sci. 3593, Springer, Berlin (2005), 144-153.

[Seilonen 2006] Seilonen, I.: "An Extended Process Automation System: An Approach based on a Multi-Agent System"; Doctoral Thesis, 2006, Helsinki University of Technology, Finland.

[Sycara 1998] Sycara, K.P.: "Multiagent Systems"; Magazine of American Association for Artificial Intelligence, 19, 2 (1998), 79-92.

[The OPC Foundation 2007] The OPC Foundation: "The OPC Unified Architecture Specifications: Parts 1-11", Version 1.xx, 2006-2007, `http://opcfoundation.org//Downloads.aspx`, Accessed online in December 2007.

[Thomas et al. 2001] Thomas, S.J., Mueller, J.B., Harvey, C.G., and Surka, D.M.: "Monitoring and Analysis of Multiple Agent Systems"; Proceedings of the NASA/JPL Workshop on Radical Agent Concepts (2001), `http://www.psatellite.com/papers/wrac2001.pdf`, Accessed online in January 2008.

[Voos 2000] Voos, H.: "Intelligent Agents for Supervision and Control: A Perspective"; Proceedings of the 15th IEEE International Symposium on Intelligent Control, IEEE Press (2000), 339-344.

[Wernstedt and Davidsson 2002] Wernstedt, F. and Davidsson, P.: "An Agent-Based Approach to Monitoring and Control of District Heating Systems"; Proceedings of the 15th International Conference on Industrial and Engineering, Applications of Artificial Intelligence and Expert Systems (IEA/AIE 2002), Lect. Notes in Artificial Intelligence 2358, Springer, Berlin (2002), 801-811.

[Wooldridge 2002] Wooldridge, M.J.: "An Introduction to Multiagent Systems"; Wiley, Chichester (2002)

[Xue et al. 2006] Xue, X., Liu, X., and Li, R.: "Towards a Customized Methodology to Develop Multi-agent Systems"; Proceedings of the 9th Pacific Rim International Workshop on Multi-Agents, Lect. Notes in Artificial Intelligence 4088, Springer, Berlin (2006), 105-116.