15th Triennial World Congress, Barcelona, Spain

USING UML FOR MODELLING REMOTE ACCESS TO MANUFACTURING SYSTEMS

Isidro Calvo, Marga Marcos, Isabel Sarachaga, Darío Orive

Department of Automatic Control and Systems Engineering
E.T.S.I. de Bilbao (University of the Basque Country)
jtpcagoi@bi.ehu.es, jtpmamum@bi.ehu.es, imvsagoi@bi.ehu.es, jtporred@bi.ehu.es

Abstract: This paper describes a general architecture to access remotely to manufacturing plants through Internet-like networks. The operations that remote users may perform over the plant are analysed in order to identify the different user profiles to be defined as well as the set of objects that constitute the so-called *Virtual Plant Model*. This Model resides in the *Application Server* which is responsible for maintaining an image of the real plant state that can be accessed by remote users. When a remote user logs on, the *Application Server* creates a *View Model* of the plant through which operations on the plant are performed. The paper discusses the use of UML diagrams that may be used in order to outline the architecture from different points of view. *Copyright © 2002 IFAC*

Keywords: Object modelling techniques. Remote Control. Distributed computer control systems. Flexible manufacturing systems. Computer-integrated manufacturing.

1. INTRODUCTION

Today's manufacturing systems tend to get increasingly complex. Moreover, new challenges come up, due to the adaptation to the constant changes in production strategies required by changes in technology and the need for new products. It is also necessary to respond to the integration problems created by the use of devices of several suppliers. Object oriented technologies are being used successfully in this kind of environments because they adapt much better to the new circumstances (see Adiga, 1993). Unfortunately, identifying the objects that compound a system is not trivial, this is why we believe it is convenient the use of methodologies that simplify this task. Once these objects have been identified for a generic system, it may be possible, to take advantage of the know-how and even to reuse the objects code.

On the other hand, as Internet is becoming popular some distributed object oriented architectures (such as CORBA, Java/RMI or DCOM, see Orfali and Harkey, 1998) are maturing becoming susceptible of being used in remote control (see Marcos, *et al*, 2001). These architectures offer an excellent integration with Internet/intranet environments allowing to use remote objects as if they were local. They also offer special services (security, events, concurrency control... etc.) that simplify the development of the applications.

In this paper we present a generic architecture to access remote plant controllers. We will analyse the remote operations that may be performed over a generic manufacturing plant in order to identify the remote objects that will be available. These objects will be arranged in different user profiles, so, every user may access just to a restricted number of objects and methods of the plant. This structure improves considerably the security of the system and, at the same time, reduces network overheads since it minimises the refreshing of information.

Unfortunately, object oriented modelling of complex systems (such as those found in manufacturing plants) is not a trivial task. This makes convenient the use of modelling languages, such as UML, in order to describe and simulate the systems. UML approaches the modelling through a small set of nearly independent views of a model and provides standard and helpful tools that describe the different aspects of a system (see Booch, *et al*., 1999; OMG, 1999). It is usually used to specify, visualise, construct and document the artifacts of a system. Moreover, it can be used as a communication vehicle between the system developers. The election of UML is justified by the fact that it is one of the most extended modelling language currently used. This is

partly due to that it aggregates several modelling languages that have been being used for some time.

The layout of the paper is as follows: section 2 presents other works that deal with remote plant operations as well as the use of UML in different application fields. Section 3 describes the generic architecture on which this work focuses, dedicating special attention to the *Application Server*, which is the key component of the architecture. In section 4 there is a description of a possible set of user profiles as well as the objects and methods that can be identified from them. Finally, in section 5, UML is used for describing all the aspects concerning the behaviour of the system.

2. RELATED WORK

Martí, *et al*. (1999) propose a framework to build remote monitoring applications based on Java/RMI. In this case the plant model is a tree like structure in which its leaves contain information about the real plant that is offered to the remote applications. This kind of models, also used in other examples (Knizak, *et al*. 1997), are valid for simple plants but they do not seem suitable for complex plants such as those found in the manufacturing field.

In Orfali and Harkey (1998) it is possible to find a discussion about the use of distributed object oriented technologies over Internet. Marcos, *et al*. (2001) provide some guidelines about how they may be used for controlling and monitoring remote processes.

There are works that use distributed object oriented technologies as applied to the manufacturing field. For instance, the work of Seinturier, *et al*. (1999) and Guyonnet, *et al*. (1997) deals with the mapping of MMS services (Manufacturing Message Specification, Valenzano, *et al.* 1992) over CORBA. They have adopted MMS as the basis of an object-oriented design of shop floor applications. This choice is justified by the fact that MMS is a standard resulting from a lot of expertise and, in consequence, it answers user needs in the manufacturing arena. Moreover, MMS itself is object oriented which simplifies the task of modelling. They use CORBA in conjunction with MMS and it seems that they complement each other reasonably well.

On the other hand, Becker, *et al*. (2000) present a methodology for developing distributed real-time object-oriented systems. Their approach provides a method for identifying possible design object architectures. They also provide tools to evaluate these object architectures allowing the identification of the 'best' architecture. This methodology uses UML-based object oriented models to describe the different architectures.

In Bordbar, *et al*. (2000) UML is used, in conjunction with Petri Nets, to design controllers for distributed manufacturing processes. In this work, conventional Statecharts are substituted by Petri Nets in order to provide analytic capabilities.

Selic (2000) describes a general framework, expressed in UML, for modelling resources and quality of service characteristics in real-time software systems.

Following these previous works the main goal of the research work presented here is to propose a whole architecture that, making use of object oriented technologies, allows remote access to manufacturing plants. In particular, this paper discusses the use of UML diagrams for modelling the whole system behaviour from different points of view.

3. PROPOSED ARCHITECTURE

3.1. General architecture

Systems are divided into the following components (see Fig. 1): plants, plant controllers, application servers and remote client applications. The plant is made out of the connected field devices. The plant controller takes care of the local behaviour of the application, as well as of providing the application server with communication to the real plant. The Application Server handles the Virtual Plant Model that is constituted by a set of objects and methods. It contains all the information about the real plant that can be accessed remotely, offering a set of methods that performs the allowed operations. The end user applications will use the objects as if they were local, hiding the tedious communication problems found in traditional applications. Finally, the application server acts as a middleman between the plant controller and the remote client applications. Its main mission is to handle the objects that will be used remotely. Obviously, it is recommended here the use of CORBA like architectures in order to provide the remote applications with objects they can use.

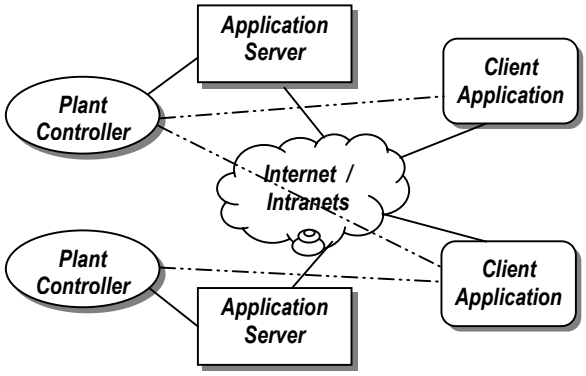

Fig. 1. General architecture.

It is important to remark that an application may be composed by several distributed plant controllers, being the Client Applications able to access them through several Application Servers, each of them containing the objects that model every Plant Controller.

3.2. Application Server

As it is depicted in figure 2 the following elements form the Application Server:

- The *Virtual Plant Model* contains the available objects that constitute the virtual plant. These objects will communicate directly to the *Plant Controller* and will provide all the operations remote users may carry out over the real plant by means of a CORBA-like distributed architecture.

- The *View Models Manager* (VM Manager) is a key element in the *Application Server*. In fact, it is an object server who is responsible for instancing the objects of the *Virtual Plant Model* and making them public for the *Client Applications*. The *View Models Manager* also creates a *View Model* when a remote user logs on, according to the permissions defined in the *Users Database*.

- The last kind of components found in the *Application Server* are the *View Models* whose mission is to connect directly to the *Client Applications* and to refresh their information. They act as filters that prevent remote users from having access to objects and methods they are not allowed to. They will contain the necessary elements that remote applications need, including information about the objects and their methods available for a specific profile, alarms that may happen in the system, pictures... etc.

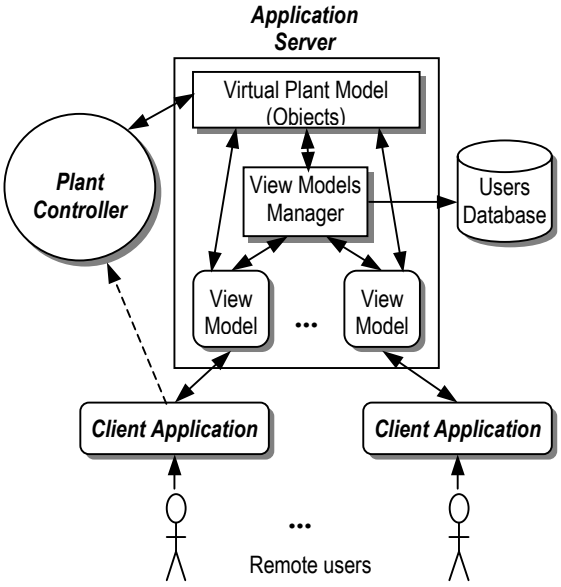

Fig. 2. Application Server and how it communicates with the rest of components of the System

This architecture will allow *Client Applications* to access the *Plant Controller* through *View Model* created by the *View Models Manager*.

4. VIRTUAL MODELLING OF MANUFACTURING SYSTEMS

This section describes the type of objects that may form the Virtual Plant Model. Firstly, some remote users with the operations they can perform over a typical manufacturing system are modelled. The next step is to identify a set of generic objects with which these operations could be carried out. This set of user profiles and objects is intended to be valid for a wide number of manufacturing plants and applications. However, the model can be enriched as necessary to adapt to new requirements. This is precisely one of the advantages of using object oriented techniques. Moreover, it is possible to re-use these objects to simplify the development of new applications.

4.1. Remote User Profiles: View Models

The following Remote User Profiles have been identified:

4.1.1. Cell Operator

This user takes direct actions over the plant. Even though he will usually be close to the plant, he is allowed to take some actions remotely. This user might keep visual contact with the plant through cameras. Following are examples of remote actions that this user could carry out over the plant:

- Management of Production Orders (typing, starting, stopping, change of priorities...)
- Manufacturing Status Monitoring
- Alarm Monitoring
- Warehouse Monitoring
- Change of Production Speed
- ...

4.1.2. Warehouse Manager

This remote user can access the warehouse contents in order to insert more raw material or withdraw manufactured items. His main tasks are:

- Warehouse Monitoring
- Withdrawal of manufactured items

4.1.3. Production Engineer

This remote user is mainly interested in the production and products. His duties involve the design and management of the programs, the improvement of the manufacturing processes... etc. Typical operations might be:

- Management of the manufacturing programs database.
- Obtaining production statistics
- Changing quality settings
- Manufacturing Status Monitoring
- ...

4.1.4. Production Manager

It is a high level user who is mainly interested in production statistics and plant performance. Typical operations are:

- Obtaining of the production statistics
- Warehouse Monitoring

4.2. The Virtual Plant: Objects and Methods

4.2.1. Warehouse

This object keeps information about the contents of the real warehouse. These are some of the operations (methods in the object oriented paradigm) that may be performed over it:

- Withdraw manufactured items
- Insert raw material
- Report warehouse contents
- ...

4.2.2. Manufacturing Status

This object allows the users to contact directly the plant controller. It will be used for monitoring purposes by keeping information about the current plant status, which devices are currently working,... etc. Additionally, it may include production orders queue objects. Example methods are:

- Report device status (one device or all)
- Change plant speed

4.2.3. Production Orders Queue

This object contacts directly the plant controller and keeps a queue with the production orders. It may be included inside the Manufacturing Status Object:

- Report orders status (status of current and pending orders)
- Insert a new production order
- Cancel a production order
- Pause a production order
- Change the priority of a production order

4.2.4. Manufacturing Programs Database Manager

This object manages the database with all the downloadable programs used to manufacture every product. It allows remote users to select the downloadable device programs used to manufacture an item:

- Add a new device program
- Delete an existing device program
- Create a new product (selects the programs used in every control device of the cell)
- Change a product (allows to change the programs used to manufacture a product)

4.2.5. Production Historic

This object allows the users to query the production history in order to obtain statistics and data related to production for a particular interval of time:

- Query the rate of good / defective products (per production order / product)
- Query the number of manufactured items (in a specific interval of time: day, month, year...)
- Query the quantity of raw material used (in a specific interval of time: day, month, year...)
- ...

5. MODELLING WITH UML

This section describes which UML diagrams can be used to model the *Application Server* which is the key system in our architecture.

5.1. User Profiles and their Remote Operations

Use case diagrams are used in UML to describe the system from the users' point of view. They are mostly used to capture the requirements of the system in the first stage of the development. In our case, they are very appropriate to identify the different user profiles, as well as the remote operations they may perform over the plant. As an example fig. 3 shows the use case diagram for the remote user *Product Engineer*.

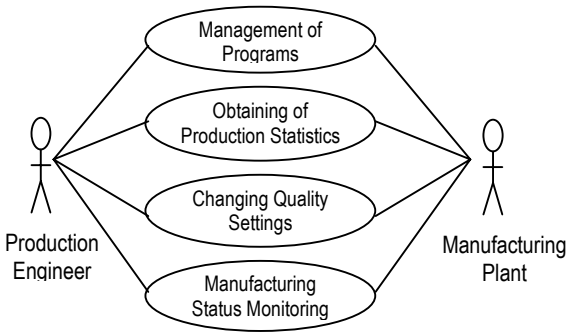

Fig. 3. Example of Use Case diagram showing the different actions the Production Engineer may take over the plant.

UML provides a template that may be used to describe every Use Case. Alternatively, this template may be substituted by plain text. Use case diagrams will describe the remote user profiles as discussed in section 4.1.

5.2. Virtual Plant Objects

The best way to depict the Virtual Plant Objects and the relationships between them is by using the UML's Class Diagram which shows the static structure and links between the different classes. This diagram provides an easy and quick way to show the hierarchy between the different classes that constitute the Virtual Plant Objects.

Figure 4 depicts an example of a simple class diagram. It shows the classes and the relationships between them. In this diagram, the Products Warehouse is a specific kind (child class) of Warehouse (parent class), the Manufacturing Status object contains a Production Queue Object and the Manufacturing Status Object uses information of one

or more Warehouses. This diagram usually shows the names of the methods for every class (not in the figure), which correspond to the remote operations that can be required from every object. Objects and methods will be based on the classes identified in section 4.2.

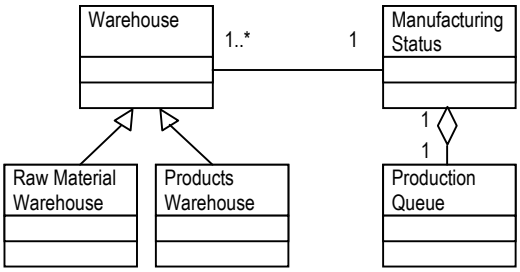

Fig. 4. Example of class diagram showing the relationship between classes.

UML also provides more diagrams related to the objects such as the Objects diagram, which shows the instances of the classes that are currently used on a system, or the Statechart diagrams which are used to outline the behaviour of the objects.

5.3. Behaviour of the Application Server

A possible way in UML to model the behaviour of the Application Server and how it interacts with the Plant Controller and the remote users is the Sequence diagram.

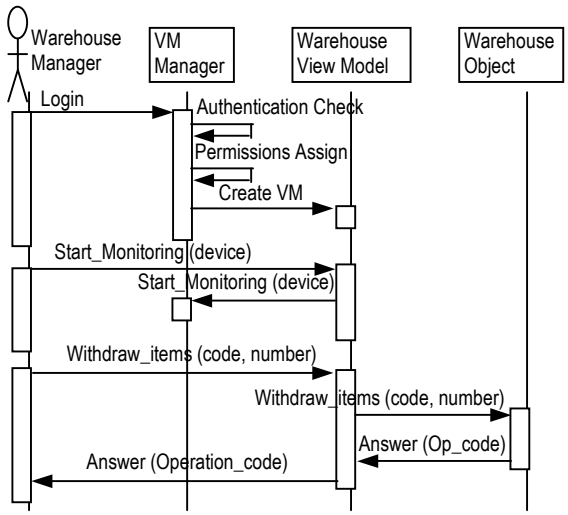

Fig. 5. Example of a Sequence diagram for Warehouse operations.

In Fig. 5 there is a diagram representing the connection of a remote user, the Warehouse Manager, to the Application Server. It also depicts the behaviour of the Application Server when this user starts the remote *Start_monitoring* and *Withdraw_manufactured_items* operations. This diagram shows the messages sent between the different elements of the Application Server. When the Warehouse Manager actor connects to the VM Manager the corresponding View Model is created. The *Withdraw_manufactured_items* operation is filtered by the Warehouse Manager View Model and, if accepted, it will be executed over the Warehouse object who will send a message to the plant controller to perform the operation.

5.4. Physical description of the system

Finally, it is also convenient to model physically the system. This is precisely the mission of the Deployment diagram of UML. This diagram shows the different computers (nodes) of the whole system as well as the necessary components required in every one of them.

In Figure 6 the Plant Controller Component (Plant Controller Node), the View Models Manager (Application Server Node) and the Client Applications (Remote PCs) will be executable files.

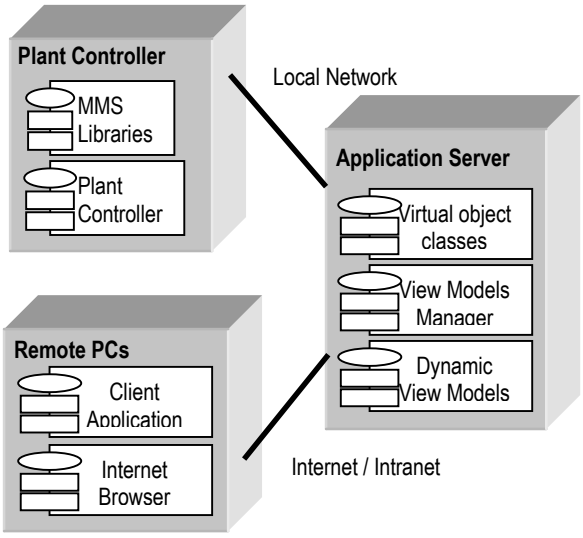

Fig. 6. Deployment diagram for the proposed architecture.

6. CASE STUDY

These ideas are being applied to a simple but real manufacturing plant in order to test their validity. The plant uses a MAP/MMS network (Valenzano *et al*. 1992) in order to simplify the building of the objects that will be compound by several MMS objects (see Fig. 7). Obviously, the same would be applicable to any other kind of physical plant with more or less effort.

The plant is constituted by several elements, as shown in Fig. 7. An intelligent warehouse, controlled by a PLC, contains both the raw material and the manufactured items. A CNC manufactures the products. A quality control centre checks the quality in order to determine which items fulfil the quality requirements. Finally, a robot acts as the transport device of the cell. Every item always follows the same path; the warehouse searches for the necessary raw item, the robot takes it to the CNC where it is manufactured and when it is finished, the robot, again, takes the item to the quality control centre. If the quality test is passed, the robot takes it back to

the warehouse where it will be stored until it is delivered, otherwise, the item will be rejected.

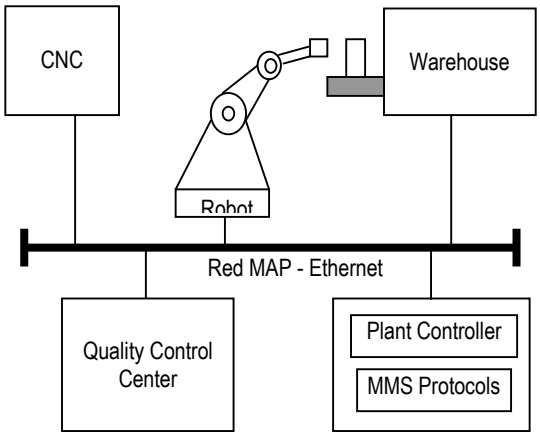

Fig. 7. Plant layout of the case study.

FUTURE WORK

The proposed architecture that has been applied to the manufacturing field can be used in different kinds of plants, such as those found in process control or other industrial plants where devices are intelligent enough to act themselves as *Application Servers*. It will also be possible to find a set of basic remote user profiles and objects that may be used on a wide number of plants. Testing the validity of the methodology for different kind of industrial plants will involve the application of these ideas to several case studies.

CONCLUSIONS

The paper presents an architecture to allow access to remote users to manufacturing plants. In order to make the architecture more flexible, several remote user profiles have been identified with the operations each of them may perform over the plant. This approach improves the security in the accesses to the plant and, at the same time, it minimises network overheads due to information refreshing.

The profiles identified pretend to be generic enough to be valid for different manufacturing systems. From them, the objects and methods to be implemented in the Application Server, the key system that provides remote access, can be identified. The interface of these objects may be reused in different applications.

Finally, it has been shown that the use of UML as the modelling language is beneficial because it allows to model the system from different points of view and provides a communication vehicle for the people involved in the development

ACKNOWLEDGEMENTS

This work has been supported by CICYT project DPI2000-1760-C03

REFERENCES

Adiga S. (1993) *"Object-Oriented Software for Manufacturing Systems"* Chapman & Hall. London.

Becker L.B., C.E. Pereira, O.P. Dias, I.M. Teixeira, J.P.Teixeira (2000) "On Identifying and Evaluating Object Architectures for Real-Time Applications" *6th IFAC Workshop on Algorithms and Architectures for Real-Time Control*, Palma de Mallorca

Booch G., J. Rumbaugh, I. Jacobson (1999) *"The Unified Modeling Language User Guide"* Addison Wesley Longman Inc.

Bordbar B., L. Giacomini, D.J. Holding (2000) "Design of Distributed Manufacturing Systems using UML and Petri Nets". *6th IFAC Workshop on Algorithms and Architectures for Real-Time Control*, Palma de Mallorca

Guyonnet G., E. Gressier-Soudan, F. Weis (1997) "COOL-MMS: a CORBA approach for ISO-MMS". *ECOOP'97 Workshop CORBA: Implementation, Use and Evaluation.* Jyvaskyla. Finland.

Knizak M., M. Kunes, M. Manninger, T. Sauter, (1997) "Applying Internet Management Standards to Fieldbus Systems", *Proc. of the 1997 IEEE International Workshop on Factory Communication Systems,* Barcelona.

Marcos M., J.M. Fuertes, I. Calvo, P. Martí, D. Orive, R. Villá, I. Sarachaga and S. Buzoianu (2001) "Object-Oriented Modeling for Remote Monitoring of Manufacturing Processes". To be presented in the *8th IEEE International Conference on Emerging Technologies and Factory Automation*, Nice

Martí P., J.C. Aguado, F. Rolando, M. Velasco, J. Colomar, J.M. Fuertes (1999) "A Java-Based Framework for distributed supervision and Control of industrial processes". *Proc. of the 7th IEEE Int. Conf. on Emerging Technologies and Factory Automation*, Barcelona

Object Management Group (1999) *"The Unified Modeling Language Specification (version 1.3)"* URL: http://www.omg.org

Orfali R., D. Harkey (1998) *"Client/Server Programming with Java and CORBA"* John Wiley & Sons

Seinturier L., A. Laurent, B. Dumant, E. Gressier-Soudan, F. Horn (1999) "A Framework for Real-Time Communication Based Object Oriented Industrial Messaging Services", *Proc. of the 7th IEEE International Conference on Emerging Technologies and Factory Automation.*, Barcelona

Selic B. (2000) "A Generic Framework for Quantitative Modeling of Real-Time Systems in UML", *Proc. of the 25th IFAC Workshop on Real-Time Programming*, Palma de Mallorca.

Valenzano A., C. Demartini, L. Ciminiera, (1992) *"MAP and TOP Communications"*, Addison Wesley.