Interactive Creation of Plant Monitors and Controls

Holger Vogelsang, Oliver Hammerschmidt, Uwe Brinkschulte, Andreas Beil
Institute for Microcomputers and Automation, University of Karlsruhe
Haid-und-Neu-Str. 7, 76131 Karlsruhe, Germany
E-mail: {vogelsang|hammer|brinks|beil}@ira.uka.de

December 11th,1996

ABSTRACT

This paper describes an approach for the interactive creation and configuration of plant monitors and controls. It is based on top of three basic services, which were built to solve several often found problems: *Fluids* for general man machine interaction, *IAM*, an intelligent measurement and control service and a platform independent real-time operating system for distributed applications. The described solution combines these services to allow an interactive configuration management and the execution of the resulting model.

Keywords: graphical user interface, data visualization, plant monitor

1 OVERVIEW

In most systems, the construction of the user interface and its connections to the application program are the most time-intensive parts in the software development. To reduce this time overhead, our institute has introduced several so called *services*, which are only configured for a special application and not modified in any way. Services are the basic unit of distribution, each service consists of at least one process. The result is a client/server based application, which uses a CORBA based system platform for communication and distribution handling as well as for process scheduling. Services provide re-usable standardized building blocks which can be created, configured, controlled and interconnected in order to construct the application, based on a service-oriented design scheme.[13]

Since most automation applications are a mixture of data acquisition, control, data storage and man-machine interaction, this paper describes an approach to combine already existing services to simplify the construction of plant monitors and controls. First, the necessary services are introduced and second, the idea of a graphical configuration is presented.

2 SYSTEM PLATFORM

The system platform offers an unique virtual operating system on top of many standard operating systems. Its most important tasks are the communication handling between the services and the process scheduling. To make the handling of distributed services transparent, a replaceable communication system is one of the basic components of the platform. This modular composition allows the exchange of the transport layer (serial line, Ethernet, ...) without any changes in the services. The API is based on CORBA.[17] The *Interface Definition Language (IDL)* is used for object access specification.[16] Object relations are expressed using CORBA's relationship specification.[15] Pictures 1 shows the structure.

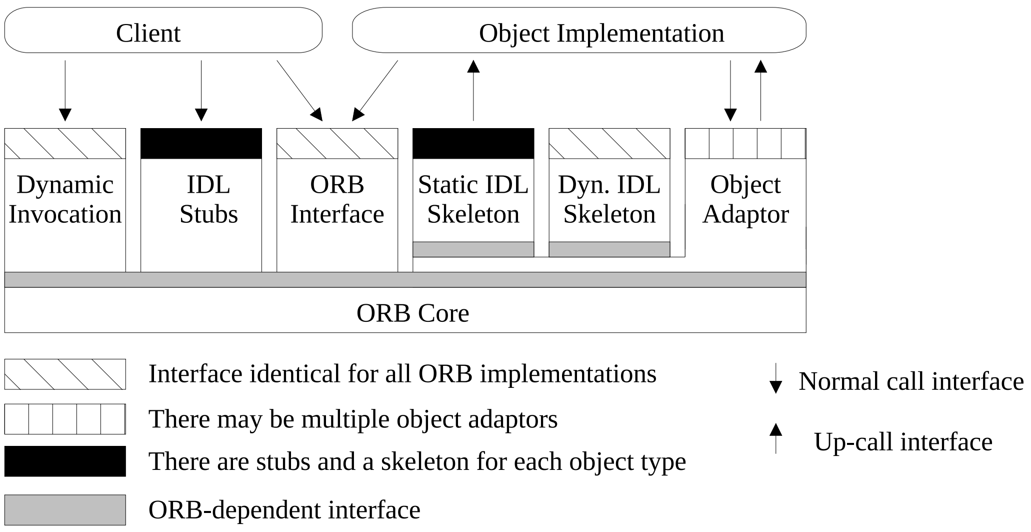


Figure 1: CORBA: Object Request Broker Structure

3 MAN MACHINE INTERACTION

The basic idea of the man machine service *Fluids*[9] is the introduction of symbols[3] as state-pictures of structured objects of an application, e.g. process variables of a control unit.[4] The user can define symbols and their behavior on events very flexible with an interactive tool, the *symbol-editor.* Symbols are composed of base-symbols, such as lines, circles and other user-defined symbols. As a result, symbols may contain a hierarchy of components. These are stored in a configuration database for usage within the application. After the configuration or construction, the symbols are available in the application. To use them, they have to be connected with an object. Changing a value of this object leads to a different graphical representation. Changing the graphical representation (e.g. the user moves a symbol interactive) leads to a different object value. The relations between object values and the resulting images can be defined. This relation is either continuous, where linear or logarithmic functions are provided, or discrete.

Fluids dynamic model allows the creation of so called bindings, which are callback methods or functions, bound to events on symbols. It is possible to link either internal functions or application defined functions to events. This mechanism is not limited to the local machine.[19]

4 MEASUREMENT AND CONTROL SERVICE

The IAM service allows to interconnect technical systems to be automated and computer-based systems for automation by intelligent actuation, measurement (data acquisition) and control of physical variables. In the area of automation and control the definition of the above mentioned basic building blocks of a service is surveyable owing to the restricted application field.[18] General measurement and actuation objects, configured along the users needs, can be combined with control transfer functions and specifications to achieve a complete complex automation system. They comprise system platforms with allocated IAM service prescriptions (soft- or hardwired) extended by converters, signal conditioning devices, actuators, sensors. The actual configuration of an IAM device depends on the functions the services should perform, e.g. a IAM device based on a microcontroller can serve only for distributed measurements of analogue physical variables whereas a IAM service based on a more powerful processor can serve for measurement, actuation and control at the same time. Well-known field-bus devices offer fixed, predefined functionalities and communication has to be realized over standardized interfaces and protocols.[12]

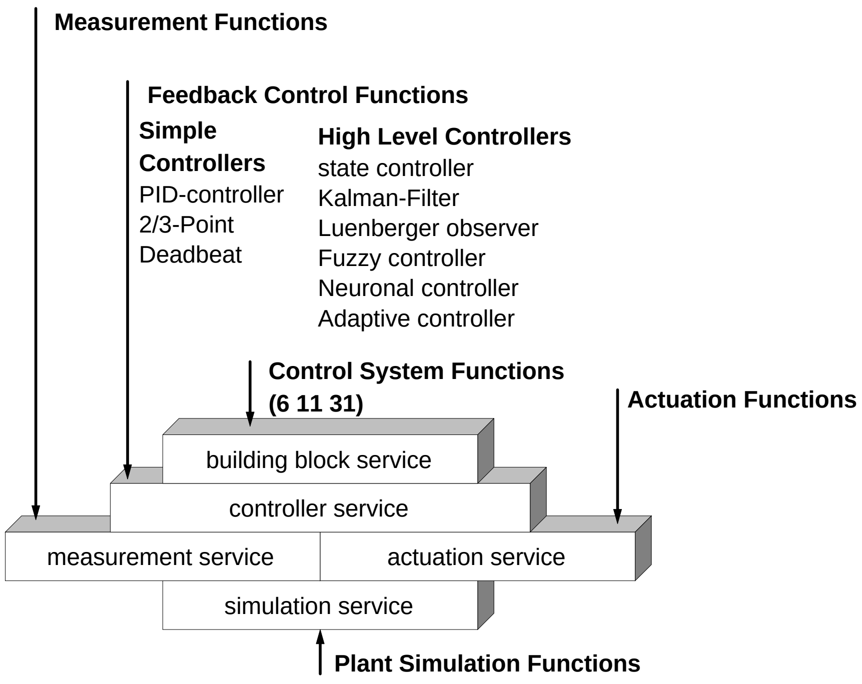


Figure 2: Layer structure of IAM

The required functionality for a service supporting control and automation applications will now be discussed in detail. The basic control services allow to interconnect technical plants to be automated and computer-based systems for automation by measurement (data acquisition), actuation, and control of physical variables (IAM-services). They comprise system platforms with allocated IAM service prescriptions (soft- or hardwired) extended by converters, signal conditioning devices, actuators, sensors. The actual configuration of an IAM device depends on the functions the services should perform, e.g. a IAM device based on a micro controller can serve only for distributed measurements of analogue physical variables whereas a IAM service based on a more powerful processor can serve for measurement, actuation and control at the same time. The functionality of the IAM services knows six functions which serve:

- to configure analogue in, digital in, analogue out, digital out, control, and surveillance agents
- to define access points with related access protocols for created IAM agents
- to initialize created IAM agents
- to interconnect access points of IAM agents with access points of other IAM and non-IAM agents
- to control agents by beginning, suspending, resuming, and ending their operation

- to place explicitly orders to agents to be carried out and to commands get results like get measured or put control values

Control agents are composed by associated measurement and actuation agents which are interconnected by building blocks with standard controllers or arbitrary transfer functions. The operations of control services are defined as a composite agent.

5 GRAPHICAL CONFIGURATION

The classical modules used in automation like converters, actuators and controllers fit into our approach representing platforms with a fixed functionality which makes part of the whole control service functionality. The explicit programming of the configuration, including access points and corresponding protocols of the interconnection and initialization of the IAM objects is tedious. Therefore tools which allow man-machine dialogue oriented configuration, test and simulation of IAM objects by a comfortable GUI do accompany all IAM services.

In order to run both IAM and Fluids in a highly efficient way, a graphical configuration, which uses an existing state picture and an IAM configuration, has been developed. It allows the description of the mutual influence of IAM and Fluids: During runtime, symbol manipulations on one hand lead to a control of the IAM operation (start measures, stop measures, change actuation values, trigger requests . . .). On the other hand, new measures result in modified symbol states to visualize the plants state. The developer is able to describe these relationships in a graphical way, using both Fluids and IAM together with the system-platform. The most important mechanism to implement these technique is the above mentioned binding of Fluids dynamical model.

5.1 Conception

This new Plant Monitor and Control Service (PMCS) has to control and monitor industrial processes of a given plant. To ensure this behavior, the service has to fulfill two tasks:

1. The states of physical components of the plant are displayed using Fluid's symbols. These symbols represent the actual state of the monitored plant in a graphical way. Relevant state changes lead to different symbol appearances (modified color, shape, movement, . . .). This behavior requires realtime capabilities of the service to ensure, that all critical states are displayed in a predictable period of time. Picture 3 shows the a typical symbol for an engine, in which the actual speed of the engine is displayed as a simple text symbol together with a graphical symbol.

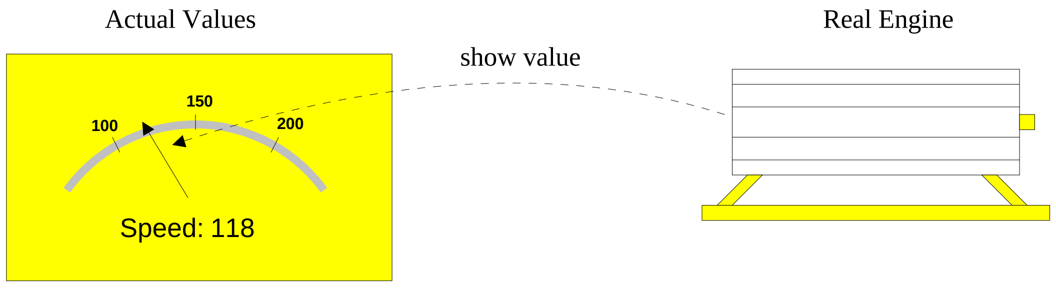


Figure 3: Display engine speed

2. On the other hand, the user of such a system has to control the states of the connected devices, using the underlying IAM service. To minimize input errors, the best way is the so called *direct manipulation* of graphical symbols: The user manipulates symbols interactively, which leads to a state change of the corresponding device. In our previous example, the users enters a new value in the text symbol or manipulates the pointer of the graphical symbol, if this is allowed by the configuration.

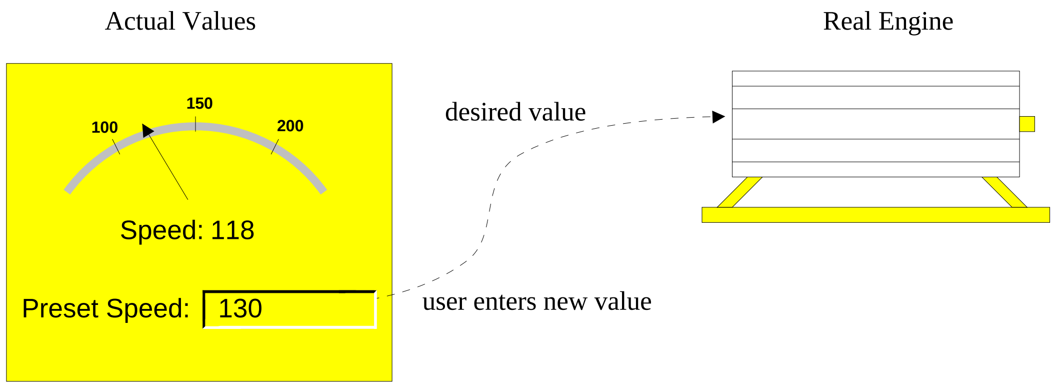


Figure 4: Modify engine speed

In many cases there is a difference between easy-to-handle values for the user and values for the devices. In this cases, the service should be able to allow precalculations or transformations between both partners. For example, some engines have to be accelerated very softly to avoid damages. This can be realized by incrementing or decrementing the value stepwise using IAM's controller functionality.

5.2 Realization

The solution consists of two parts: The Plant Monitor and Control Service itself and an interactive editor to create process pictures together with their connections to physical process variables.

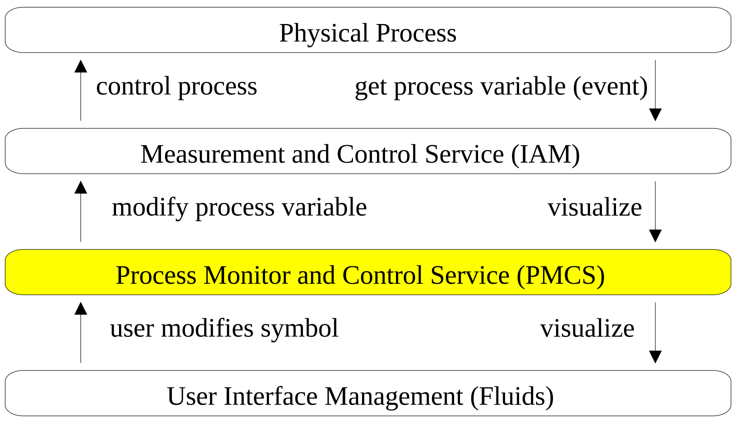
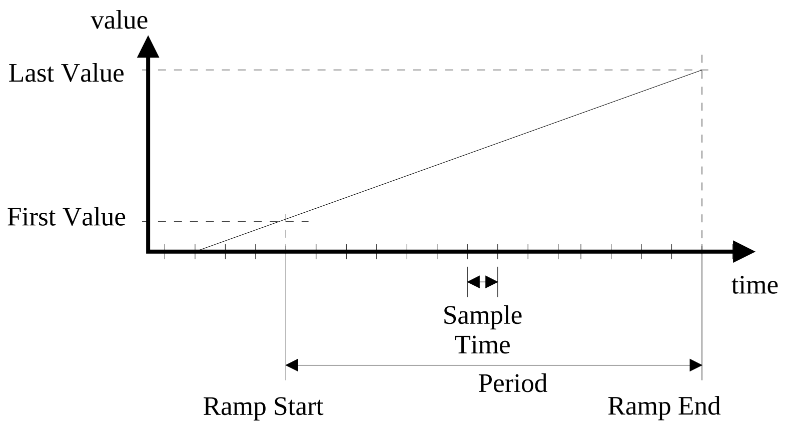
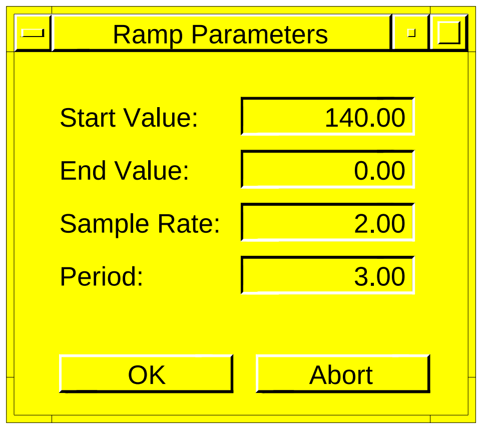

Physical Process
control process
get process variable (event)
Measurement and Control Service (IAM)
modify process variable
visualize
Process Monitor and Control Service (PMCS)
user modifies symbol
visualize
User Interface Management (Fluids)

Figure 5: Dataflow in PMCS

More complex functions bound to symbol manipulations are ramp and interval values, which are transmitted to the physical process after user interaction with symbols. Therefore, only these two functions are discussed in detail.

- **Ramp:** A ramp offers a function to increase or decrease the values of process variables in a linear way. The parameters for this kind of function are entered in a mask.

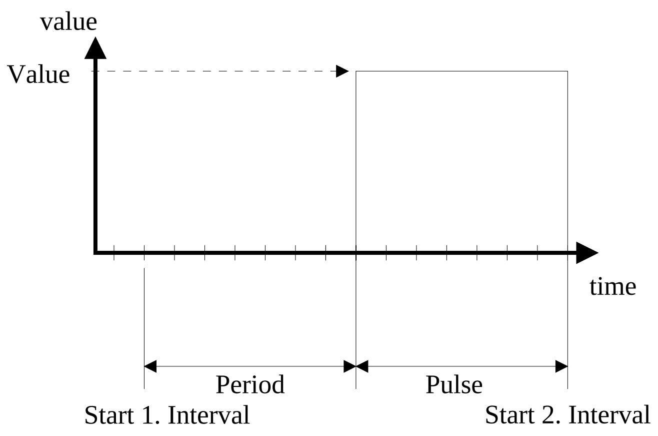


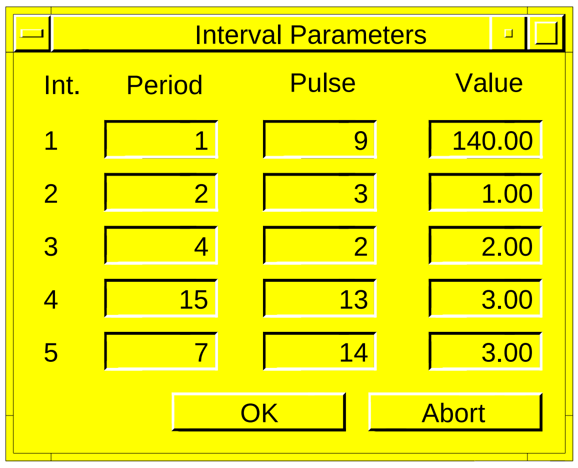

Ramp Parameters

Start Value:	140.00
End Value:	0.00
Sample Rate:	2.00
Period:	3.00

Figure 6: Ramp function and parameter input

- **Interval:** An interval function initiates a series of pulses. The parameters are entered in a mask too, a graphical input is also possible.




Interval Parameters

Int.	Period	Pulse	Value
1	1	9	140.00
2	2	3	1.00
3	4	2	2.00
4	15	13	3.00
5	7	14	3.00

Figure 7: Pulse function and parameter input

6 EXAMPLE

This example pertains to a complex application of the presented approach within process engineering. An apparatus of distillation columns is controlled and visualized for a German company of medicinal products. The system serves for development and education purposes. The control system consists of several hardware platforms which can be connected by serial links or a fieldbus (PBS). The distributed architecture is heterogeneous. One part of the processors respectively computers are used for measurement, actuation or control, some for visualization, another part for both, depending on performance and real-time constraints. Here the service-object-oriented framework shows its advantages: services and their predefined objects can be freely reconfigured and reallocated on a distributed architecture of several hardware platforms. A set of interconnected, co-operating objects describing the systems behavior is the functional specification. The implementation is done by allocating this objects to processors respectively computers. This Allocation may change during design time and partly during run-time. Different platforms and changes in communicational links are possible due to the open layer structure of the system services. So it is easy to expand an existing system with or without adding new hardware components in case of specification changes due to the open system architecture.

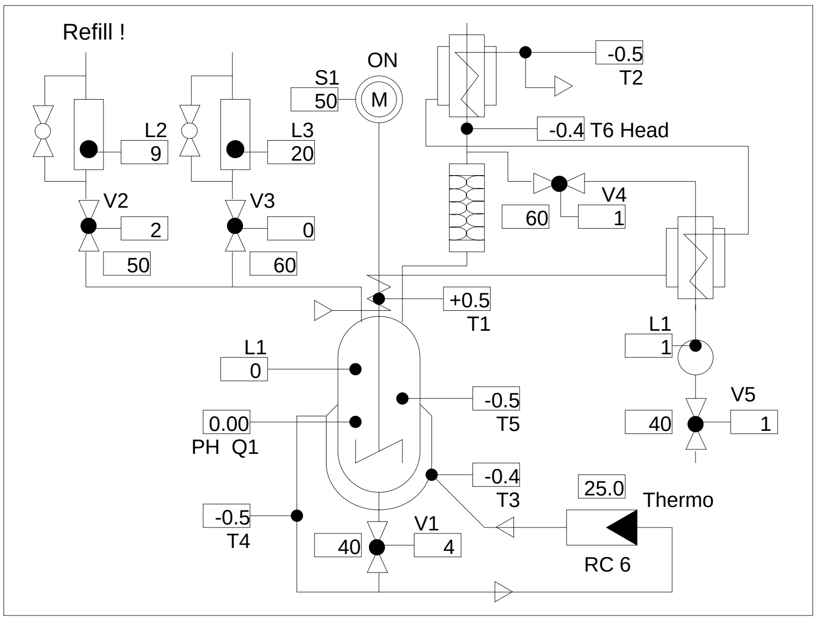


Figure 8: Distillation

7 CONCLUSIONS

Systems in automation and control are hard to handle due to real-time restrictions and intensive interaction with the environment in addition to requirements of normal medium-sized software systems like distributed functions, data and control flows. Usually the achieved implementation of the control system was time-intensive to do and is highly hardware-dependent, whereas its functional and temporal specification is not and can be given in a more abstract and comfortable graphical notation. Moreover it is well known that advantages of control systems in process automation are flexibility and the ability to implement complex hierarchies of control software easily. Here an adequate approach has to simplify this combined design process and the problems arising within its course. The presented service-oriented concept offers the possibility to construct very complex distributed control systems within a short time frame by starting from a hardware-independent specification which contains only the functional information on control structure, process sequences and graphical plant visualization. This specification can be elaborated by the use of graphical editors without knowledge of any procedural language. Changes in control parameters, controller structures, data acquisition, data storage, graphical symbols and visualization arrangements or extensions of the automation system can be done — partly also during runtime — by only using these graphical user interfaces without programming and without recompilation. All configuration information is stored in a database.

8 ANNOTATION

This paper is based on research done at the Institute for Microcomputers and Automation — Prof. Schweizer and Prof. Brinkschulte.

9 REFERENCES

[1] Len Bass, Joelle Coutaz, "Developing Software for the User Interface", *Addison-Wesley Publishing Company*, 1990

[2] U. Brinkschulte, M. Siormanolakis, H. Vogelsang, "Man Machine Service", *in: proceedings of the IAR-Workshop KEOOA'95 on Knowledge Engineering and Object Oriented Automation*, Strasbourg, France, 1995

[3] U. Brinkschulte, M. Siormanolakis, H. Vogelsang, "Graphical User Interfaces for Heterogeneous Distributed Systems", *in: proceedings of EI'96*, San Jose, USA, 1996

[4] Uwe Brinkschulte, Marios Siormanolakis, Holger Vogelsang, "Visualization and Manipulation of Structured Information", *in: proceedings of Visual'96*, February 1996, Melbourne, Australia

[5] U. Brinkschulte, H. Vogelsang, "Eine objektorientierte Schnittstelle für ein Echtzeitprozeßdatenhaltungssystem", *in: Proceedings of Echtzeit'96*, Karlsruhe, Germany, 1996

[6] U. Brinkschulte, M. Siormanolakis, H. Vogelsang, "Graphical User Interfaces for Symbol-Oriented Database Visualization and Interaction", *in: proceedings of EI'97*, San Jose, USA, 1997

[7] Hans-Jörg Bullinger, Klaus-Peter Fähnrich, Christian Janssen, "Ein Beschreibungskonzept für Dialogabläufe bei graphischen Benutzungsschnittstellen", *Informatik Forschung und Entwicklung 11: 84–93*, Springer, 1996

[8] David T. Clarke, Geoff P. Crum, "Dialogue Specification and Control: A Review of Models and Techniques", *Information and Software Technologie, Volume 9/36*, 1994

[9] Ralf Danzer, "An Object-Oriented Architecture for User Interface Management in Distributed Applications", *University of Kaiserslautern, Fachbereich Informatik*, 1992

[10] Rich McDaniel, Brad A. Myers, "Amulet's Dynamic and Flexible Prototype-Instance Object and Constraint System in C++", *Carnegie Mellon University, Human-Computer Interaction Institute Technical Report CMU-HCII-95-104*, July 1995

[11] Alan Dix, "Human-computer interaction", *Prentice Hall*, 1993

[12] D. Galara, B. Iung, G. Morel, F. Russo, "Intelligent actuation and measurement: system based modeling in priam", *in: proceedings of the 2nd IFAC Workshop on Computer Software Structures*, August 94, Lund, Sweden

[13] Oliver Hammerschmidt, Holger Vogelsang, "Design of Distributed Real-Time Systems in Process Control Applications", *in: proceedings of I-CIMPRO'96*, Eindhoven, The Netherlands, 1996

[14] R. Jacob, “A Specification Language for Direct-Manipulation User Interfaces”, *ACM Transactions on Computer Graphics, Vol. 4, No. 4, Page 283–317*

[15] OMG, “Joint Object Services Submission — Relationship Service Specification”, *Object Management Group (OMG), Technical Paper 94-05-05*, 1994

[16] OMG, “IDL C++ Language Mapping Specification”, *Object Management Group (OMG), Technical Paper 94-09-14*, 1994

[17] OMG, “The Common Object Request Broker: Architecture and Specification — Revision 2.0”, *Object Management Group (OMG), Technical Paper 95-07-20*, 1995

[18] C. Pereira, Th. Rathke, “Objektorientierte Entwicklung von Echtzeitsystemen in der Automatisierungstechnik”, *in: proceedings of the 39th Int. Wissensch. Kolloquium*, Sep. 94, Illmenau, Germany

[19] H. Vogelsang, U. Brinkschulte, M. Siormanolakis, “Flexible Dynamic Models for User Interfaces”, *in: proceedings of EI’97*, San Jose, USA, 1997