LUND UNIVERSITY

Structured Modelling of Chemical Processes with Control Systems

Nilsson, Bernt

1989

Document Version:
Publisher's PDF, also known as Version of record

Link to publication

Citation for published version (APA):
Nilsson, B. (1989). *Structured Modelling of Chemical Processes with Control Systems*. (Technical Reports TFRT-7439). Department of Automatic Control, Lund Institute of Technology (LTH).

Total number of authors:
1

General rights
Unless other specific re-use rights are stated the following general rights apply:
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.
• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal

Read more about Creative commons licenses: https://creativecommons.org/licenses/

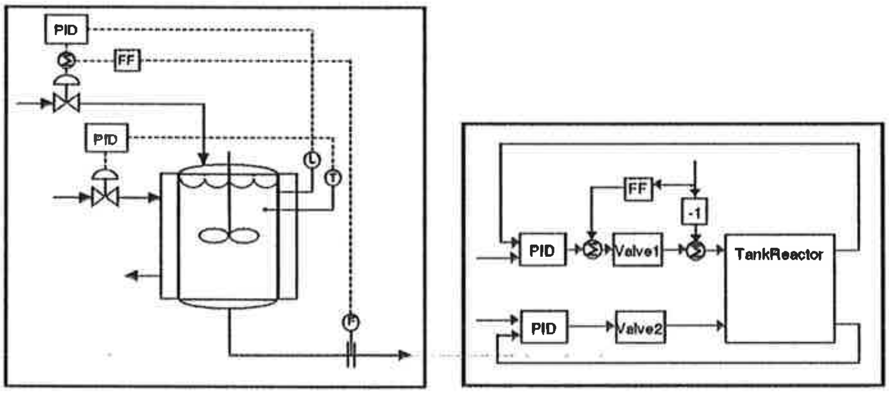
Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

LUND UNIVERSITY

PO Box 117
221 00 Lund
+46 46-222 00 00

CODEN: LUTFD2/(TFRT-7439)/1-7/(1989)

Structured Modelling of Chemical Processes with Control Systems



Bernt Nilsson

Department of Automatic Control
Lund Institute of Technology
November 1989

Department of Automatic Control
Lund Institute of Technology
P.O. Box 118
S-221 00 Lund Sweden

Document name
Report

Date of issue
November 1989

Document Number
CODEN: LUTFD2/(TFRT-7439)/1-7/(1989)

Author(s)
Bernt Nilsson

Supervisor

Sponsoring organisation
Swedish Board of Technical Development, contract 87-02503

Title and subtitle
Structured Modelling of Chemical Processes with Control Systems

Abstract

In this paper we discuss an object-oriented approach to modelling of chemical processes with control systems. The basic elements in object-oriented modelling methodology are modularization, model encapsulation, hierarchical submodel decomposition, model parameterization and inheritance. Models have an internal structure of model components, like terminals, parameters and behaviour descriptions. The model behavior can be described with equations or as a connected structure of submodels. Models and model components are represented as objects in single inheritance object class hierarchies. Chemical processes and control systems can be described with the same basic concepts. The object-oriented model representation is implemented in a prototype called System Engineering Environment, SEE. The SEE architecture allows different tools to operate on the models. Tasks that are facilitated in object-oriented modelling are model reuse and model development.

Key words
Computer simulation; computer-aided design; modeling; process control; process models.

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title

ISBN

Language
English

Number of pages
7

Security classification

Recipient's notes

The report may be ordered from the Department of Automatic Control or borrowed through the University Library 2, Box 1010, S-221 03 Lund, Sweden, Telex: 33248 lubbis lund.

Paper to be presented at the AIChE annual meeting
1989, 5-10 November, San Fransisco, CA.

Structured Modelling of Chemical Processes with Control Systems

Bernt Nilsson
Department of Automatic Control, Lund Institute of Technology
Box 118, S-221 00 Lund, Sweden
Phone +46 46 108796, Usenet: Bernt@Control.LTH.Se

Abstract. In this paper we discuss an object-oriented approach to modelling of chemical processes with control systems. The basic elements in object-oriented modelling methodology are modularization, model encapsulation, hierarchical submodel decomposition, model parameterization and inheritance. Models have an internal structure of model components, like terminals, parameters and behaviour descriptions. The model behavior can be described with equations or as a connected structure of submodels. Models and model components are represented as objects in single inheritance object class hierarchies. Chemical processes and control systems can be described with the same basic concepts. The object-oriented model representation is implemented in a prototype called System Engineering Environment, SEE. The SEE architecture allows different tools to operate on the models. Tasks that are facilitated in object-oriented modelling are model reuse and model development.

Keywords: Computer simulation; computer-aided design; modeling; process control; process models.

1. Introduction

In this paper we are going to discuss an object-oriented approach to modelling of chemical processes with control systems. Benefits of this approach are facilitated model development and model reuse. It is also possible to adapt and refine models to capture new conditions and demands.

Model structuring concepts are the key to create a modelling environment with these benefits. The models have a given internal structure of model components. An object-oriented approach to modelling represents both models and model components as objects. Modularization, decomposition, parameterization and inheritance are the basic elements in this object-oriented modelling methodology.

A new environment for system engineering (SEE) has been designed with these model structuring concepts. The basic design is composed of a model database, model/user interface and tools that operate on models. One tool, that is implemented in a prototype, is a simulator for differential and algebraic equations. This architecture allows an object-oriented approach to model development and an equation-oriented approach to the problem solving. SEE is presented in Mattsson and Andersson (1989), Andersson (1989a) and in Nilsson et al (1989).

This paper is organized as follows: An example of a process model is discussed in Section 2. Object-oriented modelling and model structuring concepts are introduced in Section 3. Modelling of controlled chemical process is discussed in Section 4 and in Section 5 are some conclusions.

2. The Tank Reactor Example

The main ideas are illustrated on a minor chemical process part, namely an exothermic continuous stirred tank reactor. The reactor is assumed to be homogeneous in concentration and temperature. A chemical reaction is assumed to occur, $A \rightarrow B$, and it produces heat. The reactor vessel

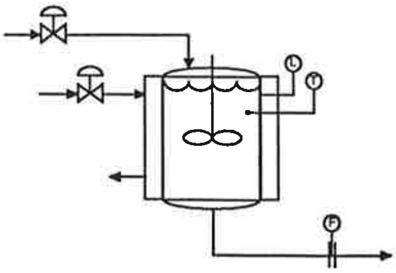


Figure 1. The continuous stirred tank reactor.

can now be modelled with a dynamic mass balance, dynamic component mass balances and an energy balance. The feed to the reactor is controlled by a valve. The outflow of the reactor is set by the surrounding system. Heat is removed by a cooling jacket. The cooling jacket can be assumed to be homogeneous and modelled by a dynamic energy balance. The cooling medium flow is also controlled by a valve.

The mathematical model of the reactor system then becomes a set of nonlinear differential equations:

$$\begin{aligned}
\rho\frac{dV}{dt} &= \rho q_{in} - \rho q_{out} \\
\frac{d(Vc)}{dt} &= q_{in}c_{in} - q_{out}c + Vr \\
\rho C_p\frac{d(VT)}{dt} &= \rho C_p q_{in} T_{in} - \rho C_p q_{out} T - Q \\
\rho_j C_{p_j}\frac{d(V_j T_j)}{dt} &= \rho_j C_{p_j} q_j T_{j_{in}} - \rho_j C_{p_j} q_j T_j + Q
\end{aligned}$$

$$r_1 = -r_2 = -k_0 e^{-\frac{E_a}{RT}} c_1 \quad ; \quad Q = \kappa A(T - T_j)$$

The concentration, c, and reaction velocity, r, are column vectors with the length of two, to describe the components A and B.

The out flow of the tank is described by a static momentum balance, which means that the flow is a function of the height in the tank and the pressure drop over the tank:

$$\rho g h + p_{tank} = \frac{(q_{out}/a)^2}{2} + p_{outlet}$$

The pressure drop over the valves can be modelled by a static momentum balance too. The pressure drop is a function of the flow and the valve position:

$$\Delta p = \frac{Ku}{2A^2} q_{in}|q_{in}| \quad ; \quad 1 \geq u \geq 0$$

The tank reactor model described above is equation-oriented. Model representations, like this, do not have any structure. The model is hard to reuse in new applications. It is not easy to change the model and it is hard to read and understand the model for a unexperienced user.

3. Object-Oriented Modelling

In object-oriented modelling models are represented as objects. Object-oriented modelling is based on the methodology from object-oriented programming. A good introduction to object-oriented programming is given in Stefik and Bobrow (1984).

An object-oriented model representation has been design in the SEE-prototype (Andersson, 1989a). A textual language for the model representation is called Omola, Object-oriented Modelling Language (Andersson, 1989b).

In this section we are first discussing some *model structuring concepts* and then the *inheritance concept*.

Internal Model Structure

A model object has an internal structure of model component objects. The internal structure of a model is composed of three major component types:

1. *Terminal* is a model component which can be used to describe interaction with a connected model.

2. *Parameter* is a model component that allows the user to interact with the model, in order to adapt its behaviour to new applications.

3. *Behaviour description* or realization is a description of the model behaviour. The behaviour can be primitive, expressing the behaviour symbolically with equations, or it can be composite and described by a structure of connected submodels. Models can have multiple realizations.

Model structuring concepts are disussed in more detail by Mattsson (1988). Model structures are also discussed in Åström and Kreutzer (1986) and in Åström and Mattsson (1987).

An object-oriented model representation of the reactor vessel can be seen in Figure 2. It is composed of terminals, parameters and primitive behaviour description. There are five terminals and seven parameters. The behaviour is described by dynamic mass, component and energy balances, which are the same as the first three differential equations in Section 2, one static momentum balance.

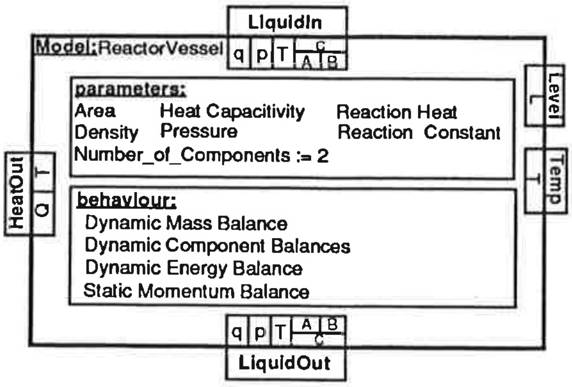


Figure 2. The internal structure of the reactor vessel model.

Submodel Interaction

In Figure 3 the structure of the reactor system is shown and it is described as a composite model object. The different parts are modularized into submodel objects. The connections between submodels represents submodel interactions. The interaction between two submodels is given by the terminal descriptions. If a connection is drawn between two terminals then the system make a consistency check. The two terminals on each side of the connection must have the same internal structure.

Terminals with internal structures, that describe a pipe connection, are the **LiquidIn** and **LiquidOut** in Figure 2. Connections can have natural interpretations, like the one between the tank reactor model, `TankReactor`, and the `Valve1` model object. In a mathematical model this connection represents a set of relations between variables. This means that flow (q), pressure (p), temperature (T) and concentration (c) in **TankReactor** and in **Valve1** are set equal or summed to zero. This kind of submodel interaction is well documented by Mattsson (1989). Terminals are defined as objects. This means that an process pipe terminal class can be a super-class of every process pipe terminal object in the process model. Terminal descriptions are therefore easy and natural to reuse.

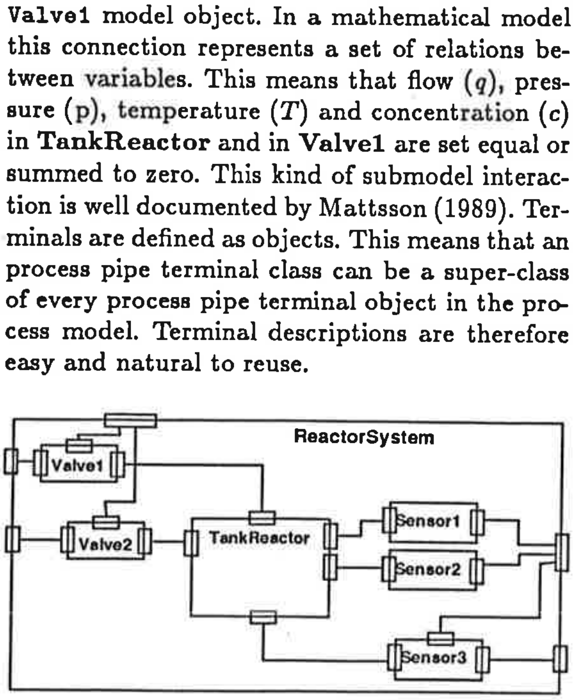


Figure 3. A block diagram showing the composite model of the tank reactor system.

Hierarchical Submodel Decomposition

The tank reactor model can be decomposed into three submodels, namely one reactor vessel model (`ReactorVessel`), one cooling jacket model (`Jacket`) and one heat transfer model (`HT-model`). The reactor vessel model is a primitive model and is seen in Figure 2. This means that the tank reactor model is a composite model, with three submodels, and we get a hierarchy of models. This is a hierarchical submodel description and it is shown in Figure 4.

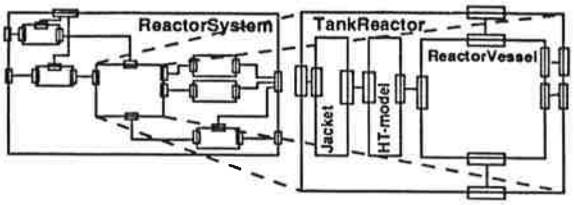


Figure 4. The hierarchical submodel decomposition in the tank reactor example.

This decomposition makes it possible to reuse submodels that is not directly interpretated as physical components. A heat transfer model object is an example of this. It is possible to change the heat transfer model without changing surrounding submodels or the super-model structure.

Inheritance

Models are represented as objects, which are subclasses of predefined super-classes. A subclass inherits properties from its super-class. The model representation has single inheritance, which means that a subclass only has one super-class. The properties that are inherited are the object attributes, which are definitions of components. Model object inherits model component definitions. A system defined super-class `Model` is the root of the model class hierarchy tree and a specialization means that attributes defining model components are added to the subclass.

An example of how to use the inheritance concept is shown in Figure 5. The class `Valve` is a subclass of the system defined super-class `Model`. It is specialized by getting two attributes that define two model components. These model components are two terminals describing the inflow and the outflow of the valve object. `Valve` is a super-class to `ControlValve`, which have two additional attributes describing the control signal terminal and a parameter. The two valves used in the reactor system are specializations of `ControlValve`. They contain specializations of the parameter attribute `Area`.

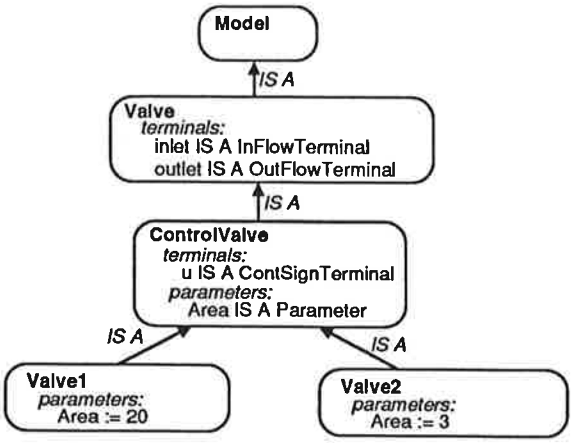


Figure 5. A part of model class hierarchy tree describing the relation between some valve models.

Parameterization

Parameterization of models and model components are important in our attempt to reuse objects. Design variables are defined as parameters, which can be changed by the user. Area and density are examples in the reactor vessel model in Figure 2. *Structure parameterization* of the reactor vessel model means that the dimension of vectors, like concentration, are being set by a parameter. The reactor vessel can be reused in a new application with another number of chemical components by changing this parameter.

An important method of parameterization is to decompose the reactor vessel model into one vessel machine model and one chemical medium model, a *medium and machine decomposition.* The machine model contains the main behaviour description (balance equations) and machine parameters (area). The medium model contains the medium behaviour (reaction velocity) and medium parameters (reaction heat and density).

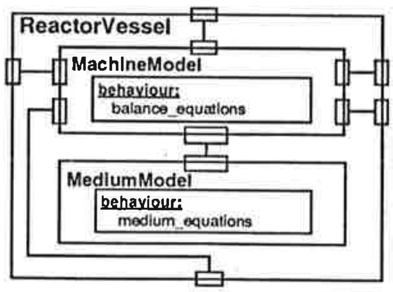


Figure 6. A medium and machine decomposed reactor vessel model.

The reactor vessel model can be decomposed into two submodels, which are connected to each other, which is seen in Figure 6. This can be seen as a parameterization of the reactor vessel. Another reactor vessel model can be created through inheritance of the attributes from the old one. The medium model can be change by overwriting the medium model definition.

A Modelling Methodology

A modelling methodology can use decomposition, parameterization and inheritance to create process models that are generic and easy to reuse.

Decomposition of process models into smallare submodels is important for abstraction of the modelling problem. Different decomposition methods are process structure decomposition into process objects, see Figure 3, transport phenomenon decomposition, like in the tank reactor model in Figure 4, or the medium and machine decomposition seen in Figure 6. The resulting submodels are often basic descriptions of fundamental behaviours.

Inheritance can be used as a model type concept and support reuse of similar models and model components. Also by overwriting inherited attributes can models be modified in order to create new models.

Parameterization of models should be made to suit the user and facilitate reused. Changing a parameter of a reused model is done by overwriting the inherited parameter value with a new value. The definition of a submodel can be changed in a similar way by overwriting the old submodel definition. One important application of this is the overwriting of media model definitions.

Decomposition and parameterization methods and the use of inheritance and discussed in Nilsson (1989).

4. A Controlled Chemical Process

We have seen how one can use an object-oriented approach to the modelling of the tank reactor process. We are now focusing on the control system description.

The Controlled Tank Reactor

The control system for the tank reactor process can be described in a similar way. The reactor has one structured terminal describing the control signal of the two valves. It has also one structured terminal describing the three sensors: level, temperature and outflow. A model of the control system is connected to the reactor system through the control signal terminal and the sensor measurement terminal. This is seen in Figure 7. The control system is a composite model with

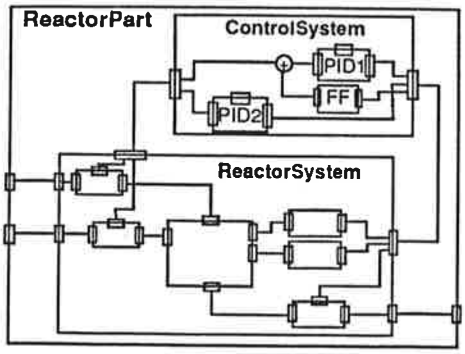


Figure 7. The tank reactor with control system.

an internal structure of submodels that represents the different controllers.

The first control system design is based on two PID-controllers, which is seen in Figure 6. One controller (PID1) uses the inflow valve to control the reactor level. The outflow measurement is used for feed forward control of the level. The other PID (PID2) controls the reactor temperature through the cooling medium valve.

Modification of the Control System

A modification of the control system is easy to do. A second control system design can be a MIMO-controller based on a LQG-design on state-space form. The state feedback and observer submodels are subclasses of generic classes with a parameterization that facilitates reuse. In this case these are specialized to capture a system with three inputs and two outputs. In an environment with tools for symbolic and numeric manipulations we can first symbolically linerize the model into a linear model and then use the numerical tool to calculate a LQG-controller, which automatically create a controller like the one in Figure 8.

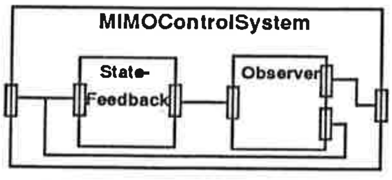


Figure 8. A state-space based MIMO control system.

This MIMO controller can now be used in the reactor part model. In a reactor part model the old definition of the control system can be overwritten by the definition of the new one. The new control system model must have terminal with the same internal structure as the old one.

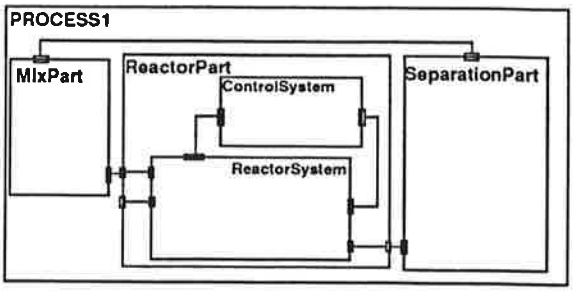


Figure 9. A chemical plant model that reuse the reactor process part model.

Plant Models

The resulting composite model, ReactorPart, can be reused in its turn in a chemical process plant model. One example is the Process1 shown in Figure 9. It is now possible to study the the control system on a complete plant model. If we have models for other parts of the process then it is easy to connect them together to create a plant model. A study of the new control system design can now be done based on realistic disturbances from the surrounding equipment.

Multiple Presentations

Large processes with control systems can be seen in a number of different ways. One way is the *process oriented view* where the controllers are distributed all over the process in order to fit the process structure description. Another way is the *control system oriented* block diagram view where the feedback loop are the most important. A third view is *computer oriented* where the hardware and software are in focus. All these views are different presentations of the same model representations. It should be possible to have different presentations of the same object.

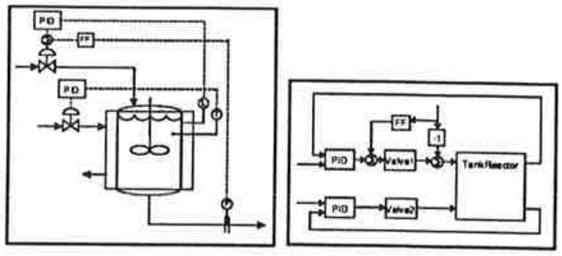


Figure 10. Two different presentations of a controlled chemical process. **Left:** a process oriented view. **Right:** a control system oriented view.

User interfaces for simulation also needs multiple presentations. A control engineer and a process operator need different interfaces. It is important to have interfaces that are natural and convenient for the user. The user interface presentations does not have to have the same structure as the model representation.

5. Conclusions

Model reuse, development, refinement and maintenance are facilitated through the concepts of modularization, decomposition, parameterization and inheritance.

Model Reuse

The strong modularization concept with encapsulated submodels with terminals supports easy and safe reuse of models. Decomposition of models into submodels makes it possible to reuse the structure and change the submodels in the structure. Advanced parameterization of models can increase the reusability of models. Inheritance means that the model object description can be distributed in a tree of super-classes and can therefore be reused.

Model Development

Model development is facilitated in three ways. One is the possibility to reuse submodels from model libraries. Predefined submodels can be reused in new composite models describing new applications. This is possible due to the strong modularization. One example is to use common process equipments, like pumps, valves etc., to create a complex process.

The possibility to decompose a process model in a multiple level description facilitate development of complex systems. The model developer can chose the amount of abstraction on each level.

Another way to facilitate model development is to use the inheritance and specialize predefined objects to describe new models in new applications. This way to develop models is of major importance and has a great potential. This is shown in the tank reactor example.

Model Refinement and Maintenance

To adapt and to modify model behaviour to real plant data requires methods for model refinement and long term use of process models requires possibilities to change, reuse and refine models. A model class can have multiple realizations and this can be used to refine the behaviour of models. A model can first get a simple behaviour description. It can then easily be refined by getting an additional behaviour description. The behaviour descriptions can be static, dynamic, simple, complex, linear or nonlinear. The user can choose a desired realization depending on the application. Model maintenance also requires readable and easierly changeable models, which are facilitated by decomposition into small objects and by inheritance.

6. Acknowledgements

I would like to thank Karl Johan Åström for initializing this work. I would also like to thank Mats Andersson and Sven Erik Mattsson, who are behind the object-oriented modelling architecture, for many and long useful discussions.

7. References

ANDERSSON, M. (1989a): "An Object-Oriented Modelling Environment," *1989 European Simulation Multiconferance, ESM'89, Rome, Italy.*

ANDERSSON, M. (1989b): "Omola - An Object-Oriented Modelling Language," Report TFRT-7417, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

ÅSTRÖM, K.J. and W. KREUTZER (1986): "System Representations," *IEEE Third Symposium on Computer-Aided Control System Design, Arlington, Virginia.*

ÅSTRÖM, K.J. and S.E. MATTSSON (1987): "High-Level Problem Solving Languages for Computer Aided Control Engineering," Report TFRT-3187, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

MATTSSON, S.E. (1988): "On Model Structuring Concepts," *4th IFAC Symposium on Computer-Aided Design in Control Systems, P.R. China.*

MATTSSON, S.E. (1989): "Modeling of Interactions between Submodels," *1989 European Simulation Multiconferance, ESM'89, Rome, Italy.*

MATTSSON, S.E. and M. ANDERSSON (1989): "An Environment for Model Development and Simulation," Report TFRT-3205, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

NILSSON, B. (1989): "Structured Modelling of Chemical Processes – An Object-Oriented Approach," Lic Tech Thesis TFRT-3203, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

NILSSON, B, S.E. MATTSSON and M. ANDERSSON (1989): "Tools for Model Development and Simulation," *Proceedings of the SAIS '89 Workshop.*

STEFIK, M. and D.G. BOBROW (1984): "Object-Oriented Programming: Themes and Variations," *The AI Magazine, Vol. 6, No. 4.*