Fourth IASTED International Symposium
on Modelling and Simulation,
June 21 - 24, 1983, Lugano, Switzerland

Combining Graphical Descriptions and Equations for Dynamical Modelling

Hilding Elmqvist

Department of Automatic Control
Lund Institute of Technology
P.O. Box 725, S-220 07 LUND 7
Sweden

ABSTRACT

The modelling of large systems is greatly simplified if their hierarchical decomposition and interconnection structure can be described graphically. The paper describes a system allowing input of block diagrams and continuous zooming to view internal details. Model equations are automatically transformed to assignment statements when possible, allowing efficient explicit integration routines to be used without additional burdens for the modeller.

1. INTRODUCTION

The structural properties of a system are very important especially when working with large models. However, these structures are hard to represent in an easily apprehendable way when a purely textual description is used for the model. The interconnection structure of submodels is, for example, much easier described graphically.

The current trend on scientific personal computers will make graphical displays much more commonly available. This will revolution the man-machine interaction. There is in particular a great opportunity to invent much more efficient problem oriented languages since, for example, the structure of large problems can be dealt with graphically.

This paper describes a system for dynamic modelling and implementation of control systems called LICS. It uses graphics for three structural properties: hierarchical decomposition, interconnection structure, and structure of interfaces. The decomposition and interconnection structure is described by block diagrams.

Model equations, declarations and control algorithms are described by text. The model equations are automatically transformed to assignment statements if an explicit integration routine is used. This is done by structural analysis of the equations using graph theoretical methods and automatic formula manipulation.

The system is designed with modern graphical scientific personal computers in mind. Joysticks are used for continuous scroll, pan and zoom of the picture in real-time. A concept of information zooming is also introduced. When, for example, zooming in on a block, it changes to a more detailed representation showing the internal structure.

2. MODELS AND GRAPHICAL REPRESENTATION

A fundamental issue when working with any complex system is its hierarchical decomposition into subsystems. A dynamic model is normally decomposed in the same way as the physical object being modelled. At each hierarchical level one has to be concerned with the interactions between submodels, i.e. what connections there are and what information is exchanged through connections.

Associated with the model structure are thus:
hierarchical decomposition structure
description of submodel interfaces
connections between submodels

We consider models basically described by:
ordinary differential equations
algebraic equations

The model language DYMOLA (Elmqvist, 1978, 1979a) introduces a textual syntax for these concepts. However, structural properties are often best conveyed using graphical representations. Since we are dealing with large models it is essential to take advantage of the possibilities for good documentation and convenient descriptions offered by graphics. Model decomposition, interface structure and connections are therefore described graphically.

The graphical representation chosen for a submodel consists of two rectangles and is shown in fig. 1. The interface description is placed at the boarder of the submodel between the inner and outer rectangle. Interfaces are described in the next section.

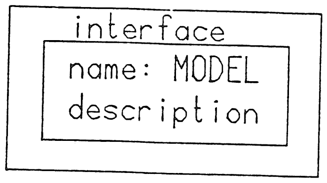

Fig. 1. Graphical form of submodel.

A primitive submodel (not further decomposed) has a textual description containing equations and declarations of local variables and parameters.

The model description of a structured submodel is a block diagram. The block diagram shows the decomposition into a set of sub-submodels and how these are connected together. Hierarchical decomposition is thus naturally described as illustrated in Fig. 2.

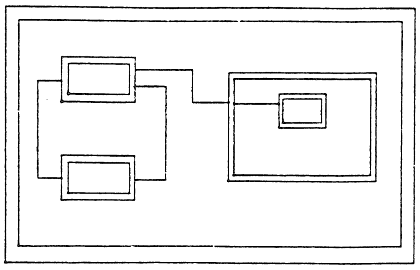

Fig. 2. Hierarchical decomposition.

3. MODEL INTERFACES AND CONNECTIONS

External influences on submodels are often concentrated to certain connection mechanisms such as shafts, pipes and electrical wires. The interaction can be modelled by introducing a set of cut variables associated with each connection mechanism. These cut variables are then used in equations within the interacting models.

It should be possible to create libraries of submodels and to use several instances of the same submodel type. This means that the submodel should have a description which is independent of its environment. The cut variables must thus be introduced in a formal manner similar to formal parameters of a procedure of a programming language. This formal parameter list describes the interaction with the calling procedure. For modelling we use several formal cut variable lists called interfaces. An interface corresponds to a connection mechanism and can be used externally in connections to interfaces of other submodels.

The graphical form of an interface will be introduced by an example.

Example:

Consider the connection mechanism of a pipe. It might be modelled by the cut variables flow rate, temperature and pressure. The graphical representation chosen is shown in Fig. 3.

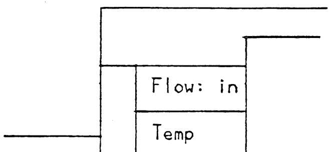

Fig. 3. Interface pipe.

The syntax for the interface components (cut variables) is:

```
name [ ':' [ 'in' | 'out' ] [ type ] ]
```

The type is real, integer or boolean. Real is default. There are two classes of variables: through and across. Through variables can be used to model flows and have the attribute 'in' or 'out'.

A connection has the graphical form of a set of connected straight lines from one interface to another. An example is shown in Fig. 4.

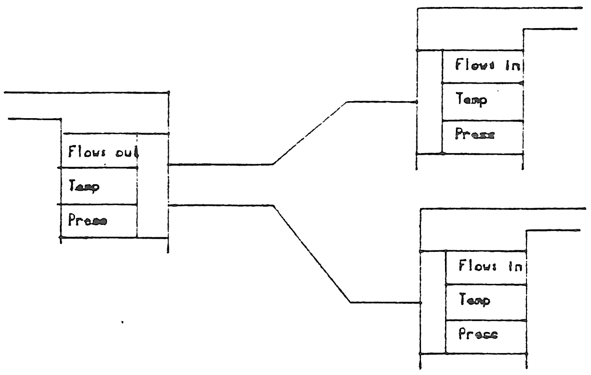

Fig. 4. Connections.

The semantics of the connections are as follows. Variables in different connected interfaces correspond to each other by position counting from left to right and from top to bottom. Corresponding variables must be of the same class and type. Corresponding across variables are considered equal. One equation is built up for each set of corresponding through variables. All in-variables are summed in the right hand part of the equation. The out-variables are summed in the left hand part. The semantics of the connection in Fig. 4. are thus:

```
M2.flow + M3.flow = M1.flow
M1.temp = M2.temp
M1.temp = M3.temp
M1.press = M2.press
M1.press = M3.press
```

Interfaces can be hierarchically structured as shown in Fig. 5.

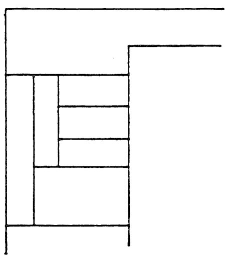

Fig. 5. Hierarchically structured interface.

Structured interfaces allow connections to be combined at higher levels of the model hierarchy. The recursive rule that corresponding structured components are connected describe the semantics of structured connections.

Connection lines terminate externally at interfaces or internally at interface components. The complete semantics of connections and interfaces is conveniently described by introducing equivalence classes of connected components.

4. MODEL EQUATIONS

The submodels at the lowest hierarchical levels are described by sets of ordinary differential equations and algebraic equations. These submodels contain, in addition to the interface descriptions, declarations of variables as locals, constants or parameters.

The equations need not be converted by the modeller to assignment statements. The form is:

expression = expression

This freedom is convenient since it corresponds to the original formulations of the models as a set of mass- and energy- balances and other basic equations. The model documentation becomes better in this way and there is no risk of introducing errors during manual transformation to assignment statements. Furthermore, the equation form is the only natural one when different types of calculations (such as simulation or calculation of operating point) are to be performed on the model or when the environments of different instances of a submodel type introduce different causality relations.

The submodel connections and the internal submodel equations constitute a set of implicit equations. Most integration routines require an algorithm that computes the derivatives and auxiliary variables. Elmqvist (1978, 1979b) discusses how the equations can be made explicit. There are three steps:

1. Choosing a unique left-hand-side variable for each equation.
2. Partitioning the equations into minimal subsets requiring simultaneous solution. Sorting the subsets into correct computational order.
3. Transforming the equations to assignment statements using automatic formula manipulation.

The first two steps are based on graph theory. The structural properties of the equations can be represented as a bipartite graph. Algorithms for step 1 and 2 can be found in Wiberg (1977) resp. Tarjan (1972). Simula implementations are found in Elmqvist (1978).

This structural analysis of the equations is of great help during modelling of large systems since it gives information about causalities and algebraic loops.

5. GRAPHICAL REPRESENTATIONS

The hierarchical properties of a model are represented by sub-block diagrams within blocks as in Fig. 2. This natural representation demands that it be possible to zoom-in a block in order to see the details.

The system allows continuous zooming, panning and scrolling (several new views each second). One joystick is used for zooming and another controls panning and scrolling. The joysticks are thus used to manoeuvre in this hierarchical space of submodels. One advantage with this spatial approach to information management is that it is very easy to learn and to use.

The need for detailed information about a block depends on the actual viewpoint. When a block diagram at one level is shown on the screen it is meaningless to show all the sub-block diagrams. It is more useful to show only their names in this case. However, when the view is changed by zooming, more detailed information should be faded in. We call this concept information zooming. Fig. 6 shows some snapshots taken during zooming of a part of a thermal power plant.

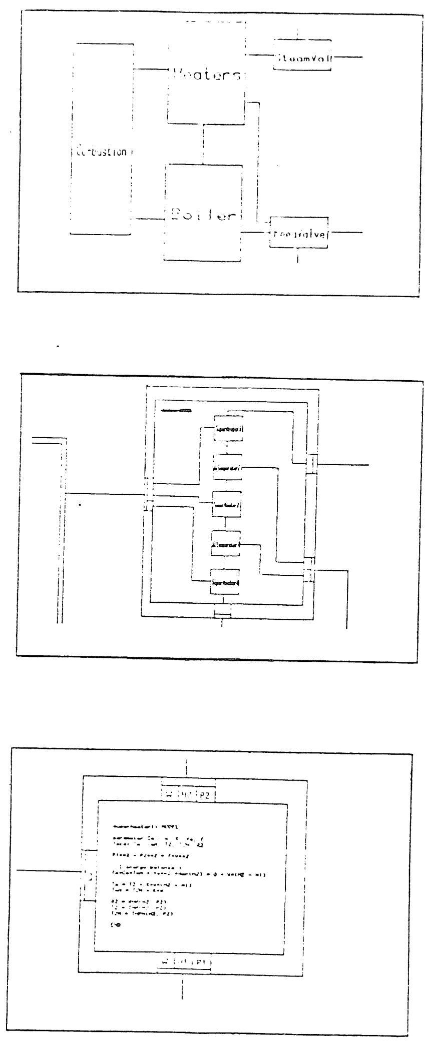
Fig 6. Zooming a thermal power plant.

The graphical system also has a windowing capability. This makes it possible to have several views of different parts of the model available on the screen at the same time.

The results of the simulations will be presented graphically. Zooming in on an interface will first give a thermometer scale showing the current value. Further zooming will give a trend curve.

The spatial approach to the handling of information has been used in other applications as well, for example in the Spatial Data Management System (SDMS) at MIT (Donelson, 1978) and in a management system for ships (Herot, Carling, Friedell, 1980). Another related work is an integrated system for program visualization (Herot, et al, 1982).

6. GRAPHICAL EDITING

Models are built up by a combined editor for text and graphics. Some of the available functions are described below. There are menu-commands to create models, interfaces, connections and text. Positioning on the screen is done using a mouse or tablet with buttons. The "model"-command stores information about how the internal details are faded in during zooming. The "interface"-command takes an interface declaration in textual form and makes a graphical layout. Text objects such as the equations of a submodel are edited by a window oriented editor.

7. CONTROL SYSTEM IMPLEMENTATION

The system is intended both for simulation and implementation of control systems (Elmqvist, 1982). The language for control algorithms is based on assignment statements with matrix notation. Separate sections of difference equations can have different sampling intervals. A "mode" concept is used to handle sequence control and starting and stopping of subsystems in a structured way.

8. IMPLEMENTATION

The system is designed with modern graphical scientific work stations in mind. The continuous zooming feature requires very fast graphical capabilities. Several complete pictures must be generated each second without flicker. We chose to base the information system on continuous zooming and information zooming assuming that the necessary hardware soon would be designed in VLSI making it cheap enough. Examples of such VLSI-designs are the graphical processors being developed at Stanford University (Clark, 1980a, 1980b).

The hardware used presently in the project is a VAX-11/780 with a frame buffer with resolution 512 x 512 x 8 attached to the Unibus.

The information system is implemented in Pascal as a set of tasks under VAX/VMS. A communication system based on the rendezvous concept of Ada has been designed.

A graphics task handles the different graphical segments and the information zooming. Zooming of characters gives a special problem. Characters generated as vector strokes gives an output rate which is too low. The character masks must be precomputed. A font generation module has been designed. It accepts font descriptions using splines (Knuth, 1979) and generates the masks of all needed sizes.

The parser for the algorithmic parts, the module for connection semantics and the execution module is currently being interfaced to the graphical system.

9. CONCLUSIONS

Advances in hardware for computer graphics mean that new kinds of man-machine interfaces are feasible and need to be explored.

The project described here explores zooming of a graphical representation of a hierarchical structure as one way of handling large structured models. Decomposition and interconnection structure are specified by block diagrams which are commonly used in engineering applications. The user merely zooms- in on a block using joysticks in order to study it in increasing detail. This spatial approach to data management is very easy to learn and use. The results of the simulations are, for example, viewed by just zooming in the variables.

The paper also outlines how the structure of the model equations can be used to convert them into an algorithm which solves for derivatives to be used together with an explicit integration routine. The user is then relieved from rewriting the equations as assignment statements.

10. ACKNOWLEDGEMENTS

The author wants to thank Tommy Essebo for his excellent implementation work and Professor Karl Johan Astrom for his support.

The work has been supported by the Swedish Board for Technical Development (STU) under contract 80-3962.

11. REFERENCES

Clark, J.H. (1980a). A VLSI Geometry Processor for Graphics. Computer, 13, 59-68.

Clark, J.H., M.R. Hannah (1980b). Distributed Processing in a High-Performance Smart Image Memory. LAMBDA, Fourth Quarter, 1980.

Donelson W. C. (1978). Spatial Management of Information. Computer Graphics, 12, 203-209.

Elmqvist H. (1978). A Structured Model Language for Large Continuous Systems. Ph.D. Thesis, TFRT-1015, Dept of Automatic Control, Lund Institute of Technology, Lund, Sweden (May 1978).

Elmqvist H. (1979). DYMOLA - A Structured Model Language for Large Continuous Systems. Proc. Summer Computer Simulation Conference, Toronto, Canada (July 1979).

Elmqvist H. (1979). Manipulation of Continuous Models Based on Equations to Assignment Statements. Proc Simulation of Systems '79, IMACS Congress 1979, Sorrento. North Holland Publ. Comp.

Elmqvist H. (1982). A Graphical Approach for Documentation and Implementation of Control Systems. Proc. 3rd IFAC/IFIP Symposium on Software for Computer Control, Madrid, Spain, (October 1982).

Herot C.F., R. Carling, M. Friedell, D. Kramlich (1980). A Prototype Spatial Data Management System. Computer Graphics, 14, 64-70.

Herot C.F., G.P. Brown, R.T. Carling, M. Friedell, D. Kramlich, R.M. Baecker (1982). An Integrated Environment for Program Visualization. Proc. Automated Tools for Information System Design and Development, New Orleans, Jan 1982.

Knuth D.E. (1979). TEX and METAFONT - New Directions in Typesetting. Digital Press, 1979.

Tarjan R.E. (1972). Depth first search and Linear Graph Algorithms. SIAM J. Comp. 1 (1972) 146-160.

Wiberg T. (1977). Permutation of an Unsymmetric Matrix to Block Triangular Form, Ph.D. Thesis, Department of Information Processing, University of Umea, Umea, Sweden (March 1977).