Experiences of Describing a Distillation Column in Some Modelling Languages

Nilsson, Bernt

1987

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

CODEN: LUTFD2/(TFRT-7362)/1-41/(1987)

Experiences of Describing a Distillation Column in some Modelling Languages.

Bernt Nilsson

Department of Automatic Control
Lund Institute of Technology
June 1987

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

Document name
Report

Date of issue
June 1987

Document Number
CODEN: LUTFD2/(TFRT-7362)/1-41/(1987)

Author(s)
Bernt Nilsson

Supervisor

Sponsoring organisation

Title and subtitle
Experiences of Desscribing a Distillation Column in some Modelling Languages.

Abstract

This report contains some experiences of implementing a physical based dynamic model of a distillation column in some modelling languages. The dynamic model chosen is a model of a binary distillation column with nine trays. The study shows that different languages handles the model structure in very different ways. Therefore the report is concentrated on how different modelling language handles the model structure. Implementations in the Simnon, Dymola and Hibliz languages are shown and discussed. Studies of some other languages is also done.

Key words
Dynamic Modelling, Distillation, Modelling Languages, Model Structure

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title

ISBN

Language
English

Number of pages
41

Recipient's notes

Security classification

Experiences of Describing a Distillation Column in some Modelling Languages.

Contents:

1. Introduction

How should a modelling language look like? This is a nontrivial problem. It depends on the problem and the user. In this report we choose a typical problem in process industry, namely modelling of a distillation column.

Implementations of the model are done in some modelling languages, which are shown and discussed. The aim with this study is to show advantages and disadvantages with existing languages. The study is focused on how the languages supports model structures, since this is very important when modelling large and complex problems.

The report is organized as follows. In Chapter 2 a description of the model is given. In Chapter 3 to 5 are the implementations in Simnon, Dymola and Hibliz discussed. In Chapter 6 some other languages are discussed. Finally a comparison is given in Chapter 7.

2. A Distillation Column Model

The aim is to implement a model of a binary distillation column with nine trays, one reboiler and one reflux drum. The feed is entering the column at tray number six. The column is controlled by two simple controllers for level control of the reboiler and the reflux drum. The models of these subsystems will be kept simple because the most interesting issue in this study is the structure of the model not the equations.

The Model

The distillation column is shown in Figure 2.1 below. The model is based on mass balances and can be found in literature (Luyben, 1973 ; Stephanopoulos, 1984). In the following subsections are each submodel described. The different submodels are the tray model, the reboiler model and the reflux drum model with the condenser.

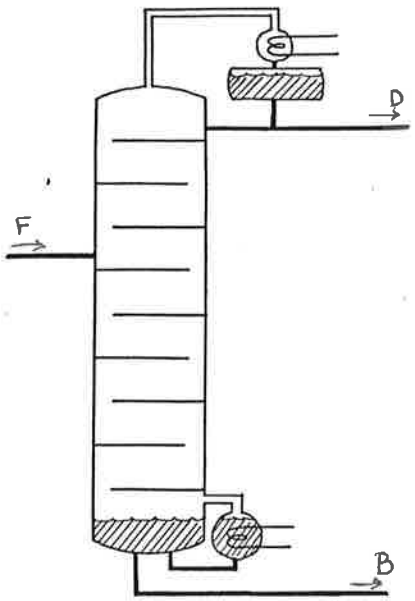

Figure 2.1 A distillation column with nine trays.

The Tray Model

The tray is only described with mass balances. A binary media is choosen for simplify the problem. The model makes following assumptions :

- Liquid and vapour leaving a tray are in equilibrium.
- Perfect mixing of the liquid.
- Neglectable vapour holdup.
- Neglectable time delays in liquid and vapour flows.
- No heat interaction with the surroundings.

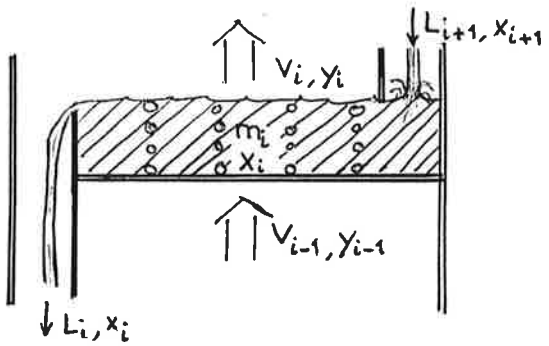

Figure 2.2 A tray.

The tray model is based on one total mass balance and one component balance:

$$\frac{dm_i}{dt} = L_{i+1} - L_i$$
$$\frac{d(m_i x_i)}{dt} = x_{i+1}L_{i+1} - x_i L_i + y_{i-1}V_{i-1} - y_i V_i$$

With neglectable vapour dynamics V_i and V_{i-1} become equal. The liquid flow from the tray will here be described with a linear expression.

$$L_i = L_{i_0} + (m_i - m_{i_0})/\beta$$

The phase equilibrium is described by relative volatility, α:

$$\alpha_{AB} = \frac{y_A/x_A}{y_B/x_B}$$

Rewriting this equation we can express vapour mole fraction y_i at tray i as:

$$y_i = \frac{\alpha x_i}{1 + (\alpha - 1)x_i}$$

The Reboiler Model

The reboiler is modelled as a tray with one mass balance and one component balance. The vapour flow through the column is created by heating in the bottom of the column. The model doesn't include energy balance but assumes that the vapour flow V_b from the reboiler is constant. The level in the reboiler is controlled by changing the bottom product flow.

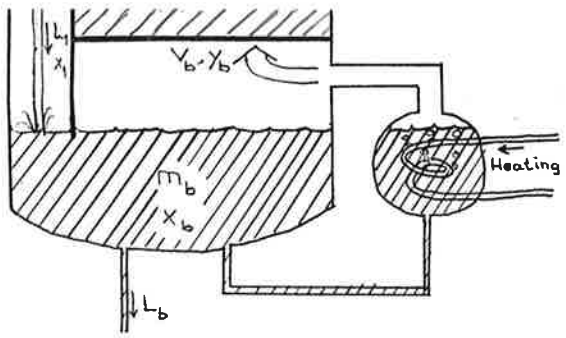

Figure 2.3 A reboiler.

The Condensation and Reflux Drum Model

The condensation is modelled as perfect with total condensing and no sub-cooling of top product. The reflux drum is modelled with one mass and one component balance. The level in the drum is controlled by changing the top product flow (distillate). The reflux flow is constant.

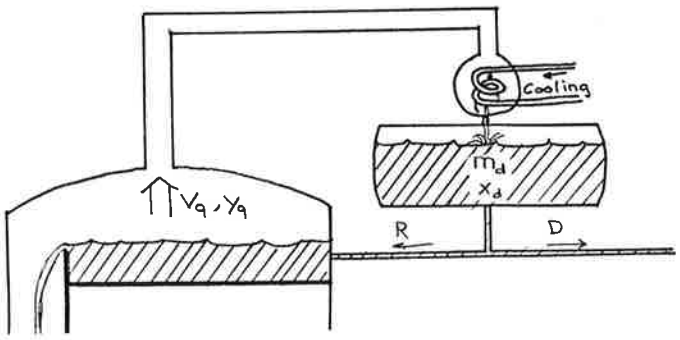

Figure 2.4 A reflux drum with a condenser.

The Level Controller

For controlling the levels in the reboiler and in the reflux drum two proportional controllers are used. To avoid large control errors the controllers has an offset parameter.

$$u = K(y_{ref} - y) + offset$$

Model Parameters

Parameters for this simple model of a distillation column was given by B. Tyreus (private communications). Below follows the parameters that was used in all simulations of the models.

Phase equilibria :	$\alpha = 2.7$
Feed conditions :	$Feedflow = 100$
	$x_{feed} = 0.9$
Reboiler flows :	$Vapourflow = 340$
	$Bottomproductflow = 10$
Flows from the reflux drum :	$Reflux = 250$
	$Destillateflow = 90$
Tray parameters :	$m_{i_0} = 20$
	$\beta = 0.25$
	$L_{i_0}(1-6) = 350$
	$L_{i_0}(7-9) = 250$

The column was initialized as follows:

Mole fraction in the hole column :	$x_i = 0.9$
Tray mass :	$m = 20$
Reboiler mass :	$m = 75$
Reflux drum :	$m = 500$

The controllers was set to give no control error at the set point

Controller gain :	$K = -2$
Reboiler control :	$y_{ref} = 75$
	$offset = 10$
Reflux drum control :	$y_{ref} = 500$
	$offset = 90$

3. Simnon.

Simnon is a special programming language for simulation of dynamical systems. Systems may be described as ordinary differential equations, as difference equations or as a combination of such equations. Simnon has an interactive environment during simulation studies (Elmqvist, 1975; Åström, 1985a and 1985b; Elmqvist et al., 1986).

The Implementation.

The implementation of the distillation column model is shown in Appendix I. It is a straightforward description in the Simnon language.

In Listing 3.1 the basic idea behind the implementation in Simnon is shown. After defining the system type (continuous system), the definition of state variable and its derivatives. Then after the definition part the model is described in a straightforward equation oriented fashion. The differential equations must be of first order and the derivative must stand on the left side of the equation. The same goes for algebraic equation where the calculated variable must stand on the left side. We can see that it is simple to implement it in Simnon, because of the repeated structure in the model description. With an advanced editor it is even more simple.

An Alternative Implementation.

In Simnon there is a concept for connecting subsystems to each other. By using the **connecting system** concept, we get a structured model for the column. The principle idea of this way of implement the model is shown in Listing 3.2. In the tray system one have to declare whats comes in and whats going out. This is made by the concepts **input** and **output**.

Unfortunate there is only one level of submodels. Therefore the model doesn't become so nice as expected. Simnon doesn't allow model types which would be a natural way of implementing our model. This means that one have to copy the tray system file nine times and give theme unique names. There are other implementation drawbacks with this way of implementing the column model which will no be discussed here.

Summary.

One of the major drawbacks in this example is that Simnon has no mechanism for repeated system description. The Simnon modelling language don't support our modelling of the distillation column with any tool for describing structures except for the connecting system concept. This has in our example a very limited power. Simnon is convenient and straightforward to use for small problems. For larger and more complicated models then the one choosen in this study the Simnon models becomes unstructured and hard to use and understand.

```
CONTINUOUS SYSTEM destil

STATE  Mb  Mxb  M1  Mx1 ...  M6  Mx6 ...  Mt  Mxt  Md  Mxd
DER   dMb dMxb dM1 dMx1 ... dM6 dMx6 ... dMt dMxt dMd dMxd

"------------------------------Column Dynamics
"-- Boiler --
dMb  = l1    - v    - botflow
dMxb = l1*x1 - v*yb - botflow*xb
xb   = Mxb/Mb
yb   = alfa*xb/(1+(alfa-1)*xb)
"-- Tray 1 --
dM1  = l2    - l1
dMx1 = l2*x2 - l1*x1 - v*(y1 - yb)
x1   = Mx1/M1
y1   = alfa*x1/(1+(alfa-1)*x1)
l1   = if M1>0 then L0s + (M1 - M0)/beta else 0
.
.
"-- Tray 6 -- (*** Feed Tray ***)
dM6  = l7    - l6                  + Feed
dMx6 = l7*x7 - l6*x6 - v*(y6 - y5) + Feed*xFeed
x6   = Mx6/M6
y6   = alfa*x6/(1+(alfa-1)*x6)
l6   = if M6>0 then L0s + (M6 - M0)/beta else 0
.
.
"-- Top Tray --
dMt  = q*reflux    - lt
dMxt =   reflux*xd - lt*xt + v*y8 - vtop*yt
xt   = Mxt/Mt
yt   = alfa*xt/(1+(alfa-1)*xt)
lt   = if Mt>0 then L0r + (Mt - M0)/beta else 0
"-- Reflux drum --
dMd  = vtop    - (reflux + destflow)
dMxd = vtop*yt - (reflux + destflow)*xd
xd   = Mxd/Md
"---------------------Physical Parameters
alfa : 2.7
.
.
END
```

Listing 3.1 A sketch of a Simnon system for the distillation column model. The complete Simnon system can be found in appendix I.

```
CONTINUOUS SYSTEM tray1

INPUT  l2 x2 vb yb
OUTPUT l1 x1 v1 y1
STATE  Mb  Mxb
DER   dMb dMxb

dM1  = l2    - l1
dMx1 = l2*x2 - l1*x1 + vb*vb*yb - v1*y1
x1   = Mx1/M1
y1   = alfa*x1/(1+(alfa-1)*x1)
l1   = if M1>0 then L0s + (M1 - M0)/beta else 0
v1   = vb

alfa : 2.7
M0   : 20
L0s  : 350
END

CONNECTING SYSTEM dest

TIME t

l1[boiler] = l1[tray1]
x1[boiler] = x1[tray1]
vb[tray1]  = vb[boiler]
yb[tray1]  = yb[boiler]

l2[tray1] = l2[tray2]
x2[tray1] = x2[tray2]
v1[tray2] = v1[tray1]
y1[tray2] = y1[tray1]
.
.
.
END
```

Listing 3.2 A sketch of an alternative way of implementing the model using connecting system.

4. Dymola.

Dymola is developed by H. Elmqvist (1978) for modelling large continuous systems. Dymola is a structured modelling language. The language allow many different submodel levels and model types. This makes it easy to create and develop model libraries.

You find the Dymola implementation of the distillation column model in Appendix IIa. Dymola could generate a Simnon system for simulation. This is shown in Appendix IIb.

Below are the most important parts of the Dymola implementation shown and discussed.

Tray Model Type.

In Listing 4.1 we can see the model type **traytype**. This is a Dymola description of the tray model in Chapter 2. For connecting the tray model with its surroundings Dymola has the concept **cut**. One tray is connected to the tray above and to the one below through the cuts **top** and **bottom**. The cut **top** is composed of the cuts **liquidin** and **vapourout**. The cut **liquidin** is in its turn composed of two variable inputs, namely **li** (liquid flow) and **xi** (composition).

```
model type traytype

cut liquidin(li,xi) vapourout(v,y)
cut vapourin(vi,yi) liquidout(l,x)
cut top [liquidin vapourout]
cut bottom [liquidout vapourin]

path liquid <liquidin - liquidout>
path vapour <vapourin - vapourout>

local m,mx
parameter l0=0, m0=0, alfa=2.7, beta=0.25

m' = li - l
mx' = xi*li - x*l + yi*vi - y*v
x = if m>0 then mx/m else 0
l = if m>0 then l0+(m-m0)/beta else 0
v = vi
y = alfa*x/(1+(alfa-1)*x)
end
```

Listing 4.1 A tray model type in Dymola.

For connecting submodels to each other, Dymola has the concept **connect**. Two submodel cuts can now be connected to each other through the command:

connect <submodel1:cut1> at <submodel2:cut2>

This way of connecting subsystems don't handles directions. Therefore Dymola has a way to describing directions. This is called **path**. In the tray model above there are two path. The path for describing the direction of the liquid flow is as following;

path liquid <liquidin - liquidout>

The two submodels can now be connected with following command;

connect (path) <submodel1> to <submodel2>.

or in our example it becomes

connect (liquid) tray2 to tray1

The power of this convention are shown in the next section.

Column Model Type.

After developing submodels of one tray, one feedtray, one reflux drum and one reboiler, the column model type can be defined. The column model is shown in Listing 4.2.

```
model type columntype

submodel (traytype) tray1,...,tray7,tray8,tray9
submodel (feedtraytype) ftray
submodel (refluxdrumtype) redrum
submodel (boilertype) boiler
.
.
connect (liquid) tray9 to tray8 to tray7 to ftray to tray5
connect (liquid) tray5 to tray4 to tray3 to tray2 to tray1
connect (vapour) tray1 to tray2 to tray3 to tray4 to tray5
connect (vapour) tray5 to ftray to tray7 to tray8 to tray9
.
.
end
```

Listing 4.2 A sketch of the column model type in Dymola. The complete model type can be found in Appendix IIa.

With the submodel statement models are created as instance of the model type. Here we create eight trays described as the model type **traytype** above.

Here we see the power of the connect concept. In the tray submodel are path for liquid and vapour flow defined. This makes it easy and comfortable to connect the trays to each other. For exemple are the liquid flow through the column described by

connect (liquid) tray9 to tray8 to tray7 ...

A Distillation Model.

The actual model are composed of one column submodel and two controller submodels which are connected as shown in Listing 4.3.

```
model distillation

submodel (columntype) destcolumn
submodel (pconttype) redrumcont boilercont

connect destcolumn:redrumlevelio at redrumcont:contio
connect destcolumn:boilerlevelio at boilercont:contio

end
```

Listing 4.3 The distillation model.

Summary.

The Dymola modelling language is very powerful and illustrative. In our example Dymola handles the model structure in a nice, intuitive way. Elmqvist (1978) proposed in his thesis an even more advanced connection mechanism. Instead of writing following;

```
submodel (traytype) tray1, tray2, tray3, ...
connect (liquid) tray9 to tray8 to tray7 ...
connect (vapour) tray1 to tray2 to tray3 ...
```

one would wont to write;

```
structured parameter n
submodel (traytype) tray[n]
for i:=1 to n-1 do
     begin
     connect (liquid) tray[i+1] to tray[i]
     connect (vapour) tray[i] to tray[i+1]
     end
```

Because of the possibility to have many different submodel levels it is interesting to declare parameters and initial condition on a desirable level. For example, if we want to change a tray parameter in the column with nine trays we have to change every one. Here we want to have a mechanism which allow us to move the parameter declaration up in the submodel hierarchy. This mean that there would be a mechanism to declare and assign parameters on a higher level, maybe implemented as the **IMPORT** and **EXPORT** concept in **Modula-2** (Elmqvist, 1978).

5. Hibliz.

Hibliz supports hierarchical block diagrams to describe the model decomposition and interconnection structure. The user can pan and zoom in the block diagram continuously. When zooming in on a block, it changes from an annotated box to a representation showing internal structure with increasing detail down to the basic equation.

The Implementation.

Hibliz uses a graphic interface for describing model structures in a convenient and intuitive way. Hibliz can be viewed as a simpler version of Dymola with an advanced graphic interface, where one uses the graphic interaction to develop the model with its submodels and their interconnections. Graphics are used for displaying cuts and connections. Creating subsystems and cuts is done by selecting commands in a menu. When creating a cut the variable must be defined as `IN` or `OUT`. This is done for knowing which sign the variable has. When this is done one can move the subsystem/cut around and place it where you want it. This is done with the mouse. Connection is also done with the mouse by selecting the connect command and click the mouse button at a cut and move the mouse to a other cut which we want to connect to the first one. In Figure 4.1 a overveiw of the Hibliz model is shown.

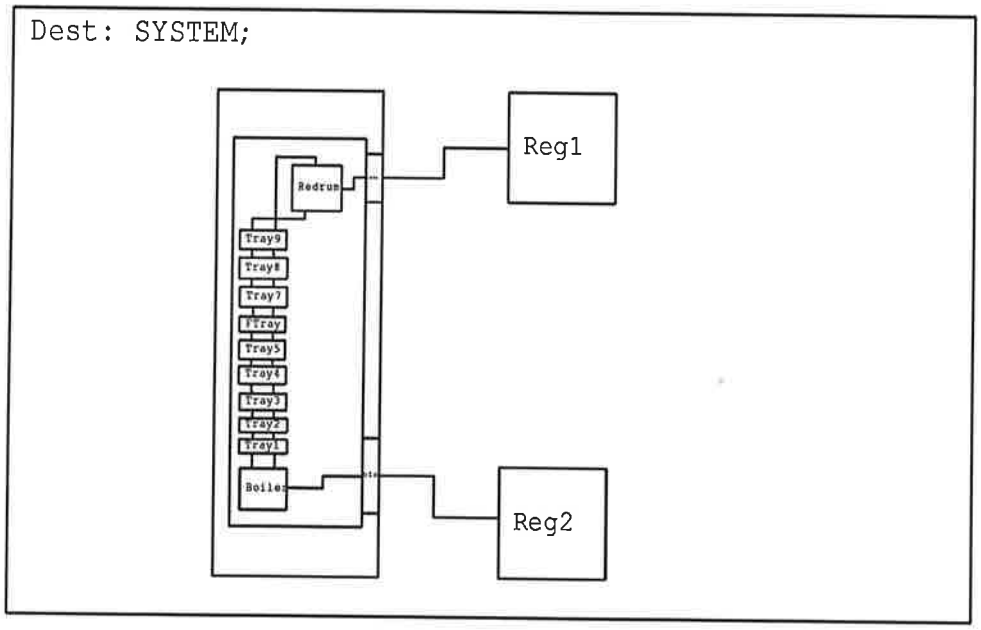

Figure 4.1 The Hibliz distillation column model.

The model description then results in a figure of the model structure. This graphic model description is saved in a file. Now the actual physical description in equations can be put in with an ordinary editor. This description of the model on equation level is straight forward. Before the equation description one have to declare parameters and states. The tray model in the Hibliz model is shown in figure 4.2 nedan

The model description with graphic interface information and equations is shown in Appendix IV.

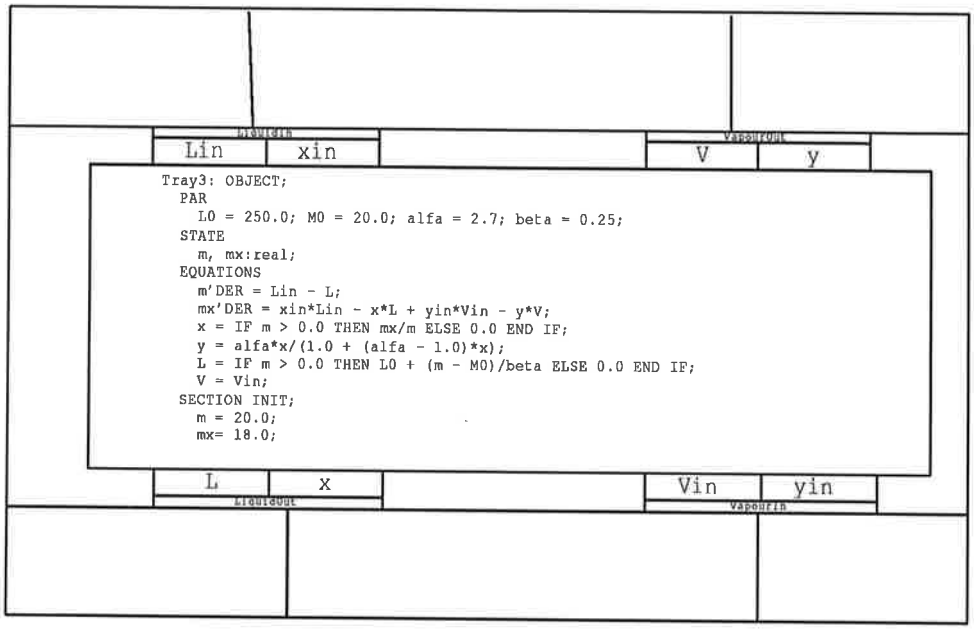

Figure 4.2 The Hibliz tray model.

Summary.

The graphic interface gives a nice interactive environment for developing models. It gives a clear overview of the model and the interconnections between its submodels. One disadvantage is that, in a graphic interface, a command is made up on a number of button pushes and some mouse movings. In a high level language this becomes quite irritating because of the low level mechanism that this gives the user to work with. To balance this there must be a number of high level commands and a lot of functions in the mouse which makes it hard to remember and time consuming to learn. A great advantage is that after developing a model one also have a graphic description of it, which in some cases is of great importance.

6. Other Modelling Languages.

In the following sections are three other languages discussed. They are all commercial available. There are no perticulair reason in the choice of languages.

Sandys.

Sandys is a program package for simulation and analysis of dynamic systems. Sandys is developed by ASEA (1985). Systems may be described by ordinary differential and algebraic equations as well as discrete events. The generic form of system description in Sandys is following:

$$f(y, y', a, t) = 0$$

Sandys allows model types. This means that it is possible to build up libraries of model types. In Sandys there is a general library for some common electrical, thermal and mechanical components. In the language there is also a concept called **FLOWS** for describing directions. This concept is similar to the **path** concept in Dymola.

The language is similar to Simnon but allow model types. Compared with Dymola, Sandys has no high level structuring concepts. This means that one can say that Sandys is some where between Simnon and Dymola in complexity. In Sandys it is possible and easy to move assignment of parameters up in the model hierarchy. This is an important feature when using model types and libraries.

ACSL.

ACSL stands for Advanced Continuous Simulation Language. ACSL is a commercial program package which is quite common in the industry (ACSL, 1986). ACSL is a **FORTRAN** preprocessor which makes the language very **FORTRAN** like.

In ACSL there is a concept called **MACRO**. This concept is very analog to the submodel concept in Dymola and the library modules in Sandys. The **MACRO** concept is an automatic code generator and not exactly a submodel concept. In the **MACRO** one have to define inputs and outputs using the **PROCEDURAL** concept. This is used as the **input/output** in Simnon. This automatic code generation has one major drawback. Lokal variable in the **MACRO** subsystem got unrecognizable names. The resulting code is very hard to understand with respect to handle model structure. In our study ACSL becomes very similar to Sandys and to summarize the ACSL study we can classify it as we classify Sandys.

Aspen Plus.

In chemical process engineering is the use of so called flowsheet simulation program quite common. Aspen Plus is a commercial flowsheeting program (ASPEN, 1984). The purpose with flowsheet simulation is to solve the stationary problem of a modelled process. This is done in an iterative fashion. The problem is to find out what comes out of the modelled process when we specify inputs and operation conditions. The use of flowsheet simulation is a way of analysing the process.

In flowsheet simulation there are no dynamics and the models are static.

In Aspen Plus you use submodels which are predefined in the language. There is no way of defining your own submodel. The language is built up around model building blocks as pumps, reactors, heaters etc.

In Aspen Plus there are separation of parameters and properties describing media and parameters describing process units. This means that the media properties are only defined ones. A lot of effort has been laid on the data base and routines for describing physical properties of media.

7. A Comparison.

As mentioned in the introduction the choice of modelling language is depending on the problem and the user. In this study we try to capture the need when modelling a large, physical based model in the process industry.

Structuring Mechanisms

To develop and work with large models it is important to have powerful mechanisms for structuring. Some basic concepts of structuring are allowing of submodels in many different levels, model types and model type libraries. Almost every one of the languages studied above have some kind of mechanisms in this direction.

The greatest differences between the languages are the connection concepts. Without competition the Dymola language has the most powerful structuring concepts. Clustering inputs and outputs in cuts, describing directions with the path concept and finally by using the connect statment one can connect submodel cuts to each other. A cut can be described with other cuts which makes the connect statment very powerful. The price Dymola have to pay for the complexity in the language is time consuming learning and how to use the language in an optimal way.

For structuring of parameter assignment there should be a concept for choosing desired submodel level for assignment. Sandys have a concept in this direction. Elmqvist (1978) proposed in his thesis a concept where parameters are defined as **externel** if they are assigned in a higher level. To have redundence this parameter is defined as **internal** on the assignment level.

Graphics

Hibliz is the only modelling language, in this study, that uses graphics. The use of block diagrams and connection lines are very illustrative in describing model structure and gives a nice overview of the hole model.

Hibliz has an analog concept to the **cut** concept in Dymola, but it has no equivalent to the **path** concept.

The connect statment becomes very natural when using graphics. In Hibliz connection is done by drawing a line with the mouse between the different cuts. This natural way of using graphics for making connections have one great disadvantage compared with Dymola. The Hibliz connection concept is very simple compared with Dymola. Things that the Dymola language handle in a high level concept can be very hard to translate to graphic based commands.

When using graphics the structure of the language becomes even more important. There is a tradeoff between command complexity and its graphic representation.

Physical approach

For large models of process industries it is convenient to make the description of the media only ones. In the flowsheeting programs there are concepts for modelling media which can be used in the process submodels.

One interesting idea when connecting submodels to each other by cuts is to allow typing of the cuts. Typing cuts in a physical sense. For example it should not be possible to connect heating media to process media or a pipe to a electric circuit.

For control purpose it would be possible to make measurements any where in the model without changing it every time one changes the measured variable. This makes it possible to have model types in libraries independent of control equipment.

8. Conclusions

A modelling language should allow submodels. For large models one should be able to have many submodel levels and possibilities to create submodels out of model types. The model type concept makes it natural to have model type libraries.

The most important and hardest problem is to develop the connection mechanisms. It must be natural to use but also powerful enought when working with large models. A natural description of connection is by cuts. A opportunity to type this cuts makes connections even more natural to work with. Advanced graphics are an important tool in making modelling language easy and illustrative to work with.

9. Acknowledgements

Many thanks to Sven Erik Mattsson for many useful discussions and instructions when working with Hibliz.

10. References

ACSL (1986): ***ACSL - Advanced Continuous Simulation Language - Reference Manual***, Fourth Edition, Mitchell and Gauthier, Assoc., Inc., Concord, Mass., USA.

ASEA (1984): "SANDYS – Användarhandledning, H6000," ASEA, Västerås, Sweden.

ASPEN (1984): ***ASPEN PLUS - Introductory Manual.***, ASPEN Technology, Inc., Cambridge, Mass..

ÅSTRÖM, K.J. (1985): "A Simnon Tutorial," Report CODEN: LUTFD2/ TFRT-3176, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

ÅSTRÖM, K.J. (1985): "Computer Aided Tools for Control System Design—A perspective," in M. Jamshidi and C.J. Herget (Eds.): ***Computer-Aided Control Systems Engineering***, North-Holland, pp. 3–40.

ELMQVIST, H. (1975): "SIMNON - User's Manual," CODEN: LUTFD2/ TFRT-3091, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

ELMQVIST, H. (1978): "A Structured Model Language for Large Continuous Systems.," Ph.D. thesis CODEN: LUTFD2/TFRT-1015, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

ELMQVIST, H. (1985): "LICS - Language for Implementation of Control Systems.," CODEN: LUTFD2/TFRT-3179, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

ELMQVIST, H., K.J. ÅSTRÖM and T. SCHÖNTHAL (1986): ***SIMNON – User's Guide for MS-DOS Computers***, Department of Automatic Control Lund Institute of Technology, Lund, Sweden.

ELMQVIST, H. and S.E. MATTSSON (1986): "A Simulator for Dynamical Systems Using Graphics and Equations for Modelling," ***Proceedings of the IEEE Control Systems Society Third Symposium on Computer-Aided Control Systems Design (CACSD)***, Arlington, Virginia, September 24–26, 1986.

LUYBEN, W.L. (1973): ***Process Modeling, Simulation and Control for Chemical Engineers.***, McGraw-Hill Book Comp., New York.

MATTSSON, S.E., H. ELMQVIST and D.M. BRÜCK (1986): "New Forms of Man-Machine Interaction," Final Report 1986-09-30, STU project 84-5069, STU program: Computer Aided Control Engineering, CACE, Report CODEN: LUTFD2/TFRT-3181, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

STEPHANOPOULOS, G. (1984): ***Chemical Process Control***, Prentice-Hall, Englewood Cliffs, New Jersey.

Appendix I

The distillation column with nine trays as a continuous Simnon system.

```
CONTINUOUS SYSTEM destil

" A simple model of a binary destillation colomn.
" The model is based on materiel balance dynamics.
" This SIMNON implementation of the model has nine trays and
" the feed is placed at tray number six.
"-----------------------------------------

STATE  Mb  Mxb  M1  Mx1  M2  Mx2  M3  Mx3  M4  Mx4  M5  Mx5  M6  Mx6
DER   dMb dMxb dM1 dMx1 dM2 dMx2 dM3 dMx3 dM4 dMx4 dM5 dMx5 dM6 dMx6
STATE  M7  Mx7  M8  Mx8  Mt  Mxt  Md  Mxd
DER   dM7 dMx7 dM8 dMx8 dMt dMxt dMd dMxd

TIME t

"-----------------------------------------Column Dynamics
"-- Boiler --
dMb  = l1    - v    - botflow
dMxb = l1*x1 - v*yb - botflow*xb

xb   = Mxb/Mb
yb   = alfa*xb/(1+(alfa-1)*xb)

"-- Tray 1 --
dM1  = l2    - l1
dMx1 = l2*x2 - l1*x1 - v*(y1 - yb)

x1   = Mx1/M1
y1   = alfa*x1/(1+(alfa-1)*x1)
l1   = if M1>0 then L0s + (M1 - M0)/beta else 0

"-- Tray 2 --
dM2  = l3    - l2
dMx2 = l3*x3 - l2*x2 - v*(y2 - y1)

x2   = Mx2/M2
y2   = alfa*x2/(1+(alfa-1)*x2)
l2   = if M2>0 then L0s + (M2 - M0)/beta else 0

"-- Tray 3 --
dM3  = l4    - l3
dMx3 = l4*x4 - l3*x3 - v*(y3 - y2)

x3   = Mx3/M3
y3   = alfa*x3/(1+(alfa-1)*x3)
l3   = if M3>0 then L0s + (M3 - M0)/beta else 0

"-- Tray 4 --
dM4  = l5    - l4
```

```
dMx4 = l5*x5 - l4*x4 - v*(y4 - y3)

x4   = Mx4/M4
y4   = alfa*x4/(1+(alfa-1)*x4)
l4   = if M4>0 then L0s + (M4 - M0)/beta else 0

"-- Tray 5 --
dM5  = l6    - l5
dMx5 = l6*x6 - l5*x5 - v*(y5 - y4)

x5   = Mx5/M5
y5   = alfa*x5/(1+(alfa-1)*x5)
l5   = if M5>0 then L0s + (M5 - M0)/beta else 0

"-- Tray 6 -- (*** Feed Tray ***)
dM6  = l7    - l6                  + Feed
dMx6 = l7*x7 - l6*x6 - v*(y6 - y5) + Feed*xFeed

x6   = Mx6/M6
y6   = alfa*x6/(1+(alfa-1)*x6)
l6   = if M6>0 then L0s + (M6 - M0)/beta else 0

"-- Tray 7 --
dM7  = l8    - l7
dMx7 = l8*x8 - l7*x7 - v*(y7 - y6)

x7   = Mx7/M7
y7   = alfa*x7/(1+(alfa-1)*x7)
l7   = if M7>0 then L0r + (M7 - M0)/beta else 0

"-- Tray 8 --
dM8  = lt    - l8
dMx8 = lt*xt - l8*x8 - v*(y8 - y7)

x8   = Mx8/M8
y8   = alfa*x8/(1+(alfa-1)*x8)
l8   = if M8>0 then L0r + (M8 - M0)/beta else 0

"-- Top Tray --
dMt  = q*reflux    - lt
dMxt =   reflux*xd - lt*xt + v*y8 - vtop*yt

xt   = Mxt/Mt
yt   = alfa*xt/(1+(alfa-1)*xt)
lt   = if Mt>0 then L0r + (Mt - M0)/beta else 0

"-- Reflux drum --
dMd  = vtop    - (reflux + destflow)
dMxd = vtop*yt - (reflux + destflow)*xd

xd   = Mxd/Md
```

```
"----------------------------------------Physical Parameters

alfa : 2.7
beta : 0.25

"----------------------------------------Operating Condition
Feed   : 100
xFeed  : 0.9

reflux : 250.6
v      : 340
q      : 1
M0     : 20

L0r    = q*reflux
L0s    = L0r + Feed
vtop   = v + (1-q)*reflux
dest0  = vtop - reflux
bot0   = Feed - dest0

"----------------------------------------Level Controllers
udf      = KD*(Md - Mdref) + dest0
destflow = if udf<0 then 0 else if udf>2*dest0 then 2*dest0 else udf
Mdref    : 500
KD       : 2

ubf      = KB*(Mb - Mbref) + bot0
botflow  = if ubf<0 then 0 else if ubf>2*bot0 then 2*bot0 else ubf
Mbref    : 75
KB       : 2

"----------------------------------------Initializing
Mb : 75
Mxb: 67.5
M1 : 20
Mx1: 18
M2 : 20
Mx2: 18
M3 : 20
Mx3: 18
M4 : 20
Mx4: 18
M5 : 20
Mx5: 18
M6 : 20
Mx6: 18
M7 : 20
Mx7: 18
M8 : 20
Mx8: 18
Mt : 20
```

```
Mxt: 18
Md : 500
Mxd: 450

END
```

Appendix II A

The distillation column with nine trays described in the Dymola language.

```
model type traytype

{ A simple model of a tray in a
  binary distillation column.  }

cut liquidin(li,xi) vapourout(v,y)
cut vapourin(vi,yi) liquidout(l,x)
cut top    [liquidin vapourout]
cut bottom [liquidout vapourin]

path liquid <liquidin - liquidout>
path vapour <vapourin - vapourout>

local m,mx
parameter l0=0, m0=20, alfa=2.7, beta=0.25

m'  =    li -   l
mx' = xi*li - x*l + yi*vi - y*v

x    = if m>0 then mx/m else 0
l    = if m>0 then l0 + (m-m0)/beta else 0

v    = vi
y    = alfa*x / (1+(alfa-1)*x)

end
```

```
model type feedtraytype

{ A simple model of a feed tray
  in a binary distillation column.  }

cut liquidin(li,xi) vapourout(v,y)
cut vapourin(vi,yi) liquidout(l,x)
cut feed(qf,xf)
cut top    [liquidin vapourout]
cut bottom [liquidout vapourin]

path liquid <liquidin - liquidout>
path vapour <vapourin - vapourout>

local m,mx
parameter l0=0, m0=20, alfa=2.7, beta=0.25

m'  =    li -   l              +    qf
mx' = xi*li - x*l + yi*vi - y*v + xf*qf
```

```
x   = if m>0 then mx/m else 0
l   = if m>0 then l0 + (m - m0)/beta else 0

v   = vi
y   = alfa*x / (1+(alfa-1)*x)

end
```

```
model type boilertype

{ A simple model of a reboiler in
  a binary distillation column.  }

cut liquidin(li,xi) vapourout(qv,y)
cut bottom(qb)
cut top [liquidin vapourout]

cut input(qb) output(h)
cut levelio [input output]
path control <input - output>

local m,mx,x
parameter alfa=2.7, area=1

m'  =    li -   qb -   qv
mx' = xi*li - x*qb - y*qv

x   = if m>0 then mx/m else 0
y   = alfa*x / (1+(alfa-1)*x)
h   = m/area

end
```

```
model type refluxdrumtype

{ A simple model of a reflux drum
in a binary distillation column. }

cut vapourin(vi,yi) liquidout(qr,xd)
cut top(qd)
cut bottom [liquidout vapourin]

cut input(qd) output(h)
cut levelio [input output]
path control <input - output>

local m,mx
parameter area=1
```

```
m'  =    vi -    qd -    qr
mx' = yi*vi - xd*qd - xd*qr

xd  = if m>0 then mx/m else 0
h   = m/area

end
```

```
model type columntype

submodel (traytype) tray1,tray2,tray3,tray4,tray5,tray7,tray8,tray9
submodel (feedtraytype) ftray
submodel (refluxdrumtype) redrum
submodel (boilertype) boiler

cut rinput(rin) routput(rout)
cut redrumlevelio [rinput routput]
cut binput(bin) boutput(bout)
cut boilerlevelio [binput boutput]

parameter qf=100, xf=0.9, qr=250, qv=340

connect redrum:bottom at tray9:top

connect (liquid) tray9 to tray8 to tray7 to ftray to tray5
connect (liquid) tray5 to tray4 to tray3 to tray2 to tray1

connect tray1:bottom  at boiler:top

connect (vapour) tray1 to tray2 to tray3 to tray4 to tray5
connect (vapour) tray5 to ftray to tray7 to tray8 to tray9

connect redrumlevelio at redrum:levelio
connect boilerlevelio at boiler:levelio

ftray.qf = qf
ftray.xf = xf

redrum.qr = qr

boiler.qv = qv

end
```

```
model type pconttype

cut input(y) output(u)
cut contio [output input]
path control <input - output>
```

```
parameter k=2, os=0, yref=1

u = k*(yref - y) + os

end
```

```
model distillation

submodel (columntype) destcolumn
submodel (pconttype) redrumcont boilercont

connect destcolumn:redrumlevelio at redrumcont:contio
connect destcolumn:boilerlevelio at boilercont:contio

end
```

Appendix II B

A modified Simnon system from the Dymola translator.

```
CONTINUOUS SYSTEM destsyst
STATE tray1_m tray1_mx ftray_m ftray_mx redrum_m redum_mx boiler_m boier_mx
DER t1_der1m t1_dermx fy_der1m fy_dermx rm_der1m rm_dermx br_der1m br_dermx
STATE tray2_m tray2_mx tray3_m tray3_mx tray4_m tray4_mx tray5_m tray5_mx
DER t2_der1m t2_dermx t3_der1m t3_dermx t4_der1m t4_dermx t5_der1m t5_dermx
STATE tray7_m tray7_mx tray8_m tray8_mx tray9_m tray9_mx
DER t7_der1m t7_dermx t8_der1m t8_dermx t9_der1m t9_dermx
tray1_l0:0
tray1_m0:20
tr1_alfa:2.7
tr1_beta:0.25
ftray_l0:0
ftray_m0:20
fty_alfa:2.7
fty_beta:0.25
rem_area:1
bor_alfa:2.7
bor_area:1
tray2_l0:0
tray2_m0:20
tr2_alfa:2.7
tr2_beta:0.25
tray3_l0:0
tray3_m0:20
tr3_alfa:2.7
tr3_beta:0.25
tray4_l0:0
tray4_m0:20
tr4_alfa:2.7
tr4_beta:0.25
tray5_l0:0
tray5_m0:20
tr5_alfa:2.7
tr5_beta:0.25
tray7_l0:0
tray7_m0:20
tr7_alfa:2.7
tr7_beta:0.25
tray8_l0:0
tray8_m0:20
tr8_alfa:2.7
tr8_beta:0.25
tray9_l0:0
tray9_m0:20
tr9_alfa:2.7
tr9_beta:0.25
desmn_qf:100
desmn_xf:0.9
desmn_qr:250
```

```
desmn_qv:340
redont_k:2
rednt_os:0
ret_yref:1
boiont_k:2
boint_os:0
bot_yref:1

"    Submodel: destcolumn::boiler
br_der1m = tray1_l - bin - desmn_qv
br_dermx = tray1_x*tray1_l - boiler_x*bin - tray1_yi*desmn_qv
boiler_x = if boiler_m > 0 then boier_mx/boiler_m else 0
bout = boiler_m/bor_area

"    Submodel: boilercont
bin = boiont_k*(bot_yref - bout) + boint_os

"    Submodel: destcolumn::tray1
tray1_yi = bor_alfa*boiler_x/(1 + (bor_alfa - 1)*boiler_x)
t1_dermx = tray2_x*tray2_l - tray1_x*tray1_l + tray1_yi*desmn_qv - h1
t1_der1m = tray2_l - tray1_l
tray1_l = if tray1_m > 0 then tray1_l0+(tray1_m - tray1_m0)/tr1_beta else 0
tray1_x = if tray1_m > 0 then tray1_mx/tray1_m else 0
tray1_y = tr1_alfa*tray1_x/(1 + (tr1_alfa - 1)*tray1_x)

"    Submodel: destcolumn::tray2
t2_der1m = tray3_l - tray2_l
t2_dermx = tray3_x*tray3_l - tray2_x*tray2_l + tray1_y*desmn_qv - h2
h2 = tray2_y*  desmn_qv
tray2_l = if tray2_m > 0 then tray2_l0+(tray2_m - tray2_m0)/tr2_beta else 0
tray2_x = if tray2_m > 0 then tray2_mx/tray2_m else 0
h1 = tray1_y*  desmn_qv
tray2_y = tr2_alfa*tray2_x/(1 + (tr2_alfa - 1)*tray2_x)

"    Submodel: destcolumn::tray3
t3_der1m = tray4_l - tray3_l
t3_dermx = tray4_x*tray4_l - tray3_x*tray3_l + tray2_y*desmn_qv - h3
h3 = tray3_y*  desmn_qv
tray3_l = if tray3_m > 0 then tray3_l0+(tray3_m - tray3_m0)/tr3_beta else 0
tray3_x = if tray3_m > 0 then tray3_mx/tray3_m else 0
tray3_y = tr3_alfa*tray3_x/(1 + (tr3_alfa - 1)*tray3_x)

"    Submodel: destcolumn::tray4
t4_der1m = tray5_l - tray4_l
t4_dermx = tray5_x*tray5_l - tray4_x*tray4_l + tray3_y*desmn_qv - h4
h4 = tray4_y*  desmn_qv
tray4_l = if tray4_m > 0 then tray4_l0+(tray4_m - tray4_m0)/tr4_beta else 0
tray4_x = if tray4_m > 0 then tray4_mx/tray4_m else 0
tray4_y = tr4_alfa*tray4_x/(1 + (tr4_alfa - 1)*tray4_x)

"    Submodel: destcolumn::tray5
```

```
t5_der1m = ftray_l - tray5_l
t5_dermx = ftray_x*ftray_l - tray5_x*tray5_l + tray4_y*desmn_qv - h5
h5 = tray5_y*  desmn_qv
tray5_l = if tray5_m > 0 then tray5_l0+(tray5_m - tray5_m0)/tr5_beta else 0
tray5_x = if tray5_m > 0 then tray5_mx/tray5_m else 0
tray5_y = tr5_alfa*tray5_x/(1 + (tr5_alfa - 1)*tray5_x)

"    Submodel: destcolumn::ftray
fy_der1m = tray7_l - ftray_l + desmn_qf
fy_dermx = tray7_x*tray7_l - ftray_x*ftray_l + tray5_y*desmn_qv - hf
hf = ftray_y*  desmn_qv - desmn_xf*desmn_qf
ftray_x = if ftray_m > 0 then ftray_mx/ftray_m else 0
ftray_y = fty_alfa*ftray_x/(1 + (fty_alfa - 1)*ftray_x)
ftray_l = if ftray_m > 0 then ftray_l0+(ftray_m - ftray_m0)/fty_beta else 0

"    Submodel: destcolumn::tray7
t7_der1m = tray8_l - tray7_l
t7_dermx = tray8_x*tray8_l - tray7_x*tray7_l + ftray_y*desmn_qv - h7
h7 = tray7_y*  desmn_qv
tray7_l = if tray7_m > 0 then tray7_l0+(tray7_m - tray7_m0)/tr7_beta else 0
tray7_x = if tray7_m > 0 then tray7_mx/tray7_m else 0
tray7_y = tr7_alfa*tray7_x/(1 + (tr7_alfa - 1)*tray7_x)

"    Submodel: destcolumn::tray8
t8_der1m = tray9_l - tray8_l
t8_dermx = tray9_x*tray9_l - tray8_x*tray8_l + tray7_y*desmn_qv - h8
h8 = tray8_y*  desmn_qv
tray8_l = if tray8_m > 0 then tray8_l0+(tray8_m - tray8_m0)/tr8_beta else 0
tray8_x = if tray8_m > 0 then tray8_mx/tray8_m else 0
tray8_y = tr8_alfa*tray8_x/(1 + (tr8_alfa - 1)*tray8_x)

"    Submodel: destcolumn::tray9
t9_der1m = desmn_qr - tray9_l
t9_dermx = xd*desmn_qr - tray9_x*tray9_l + tray8_y*desmn_qv - h9
h9 = redum_yi*  desmn_qv
tray9_l = if tray9_m > 0 then tray9_l0+(tray9_m - tray9_m0)/tr9_beta else 0
tray9_x = if tray9_m > 0 then tray9_mx/tray9_m else 0

"    Submodel: destcolumn::redrum
rm_der1m = desmn_qv - rin - desmn_qr
rm_dermx = redum_yi*desmn_qv - xd*rin - xd*desmn_qr
redum_yi = tr9_alfa*tray9_x/(1 + (tr9_alfa - 1)*tray9_x)
xd = if redrum_m > 0 then redum_mx/redrum_m else 0
rout = redrum_m/rem_area

"    Submodel: redrumcont
rin = redont_k*(ret_yref - rout) + rednt_os

END
```

Appendix III

The distillation column with nine trays described in the Hibliz language. The text file.

```
Dest: SYSTEM
 AT(-1.25000E-01,-1.25000E-01) SIZE (1.25000E+00, 1.25000E+00);
 SUBMODEL
  Column: SYSTEM
   AT( 2.75000E-01, 1.60000E-01) SIZE ( 1.35000E-01, 6.27500E-01);
   INTERFACE
    rdio :
     AT( 9.00000E-01,  7.70000E-01) SIZE ( 1.00000E-01,  1.00000E-01);
    bio :
     AT( 9.00000E-01,  1.30000E-01) SIZE ( 1.00000E-01,  1.55000E-01);

   SUBMODEL
    Tray1: OBJECT
     AT( 1.52500E-01,  2.50000E-01) SIZE ( 2.90000E-01,  3.00000E-02);
     INTERFACE
      LiquidIn: (Lin: IN real, xin: IN real)
       AT( 1.60000E-01,  9.00000E-01) SIZE ( 2.15000E-01,  1.00000E-01);
      VapourOut: (V: OUT real, y: OUT real)
       AT( 6.30000E-01,  9.00000E-01) SIZE ( 2.12500E-01,  1.00000E-01);
      LiquidOut: (L: OUT real, x: OUT real)
       AT( 1.62500E-01,  0.00000E+00) SIZE ( 2.17500E-01,  1.00000E-01);
      VapourIn: (Vin: IN real, yin: IN real)
       AT( 6.30000E-01,  0.00000E+00) SIZE ( 2.20000E-01,  1.00000E-01);

     PAR
       L0 = 250.0; M0 = 20.0; alfa = 2.7; beta = 0.25;
     STATE
       m, mx:real;
     EQUATIONS
       m'DER = Lin - L;
       mx'DER = xin*Lin - x*L + yin*Vin - y*V;
       x = IF m > 0.0 THEN mx/m ELSE 0.0 END IF;
       y = alfa*x/(1.0 + (alfa - 1.0)*x);
       L = IF m > 0.0 THEN L0 + (m - M0)/beta ELSE 0.0 END IF;
       V = Vin;
     SECTION INIT;
       m = 20.0;
       mx= 18.0;
     END;

    Tray2: OBJECT
     AT( 1.55000E-01,  2.95000E-01) SIZE ( 2.85000E-01,  3.00000E-02);
     INTERFACE
      LiquidIn: (Lin: IN real, xin: IN real)
       AT( 1.60000E-01,  9.00000E-01) SIZE ( 2.15000E-01,  1.00000E-01);
      VapourOut: (V: OUT real, y: OUT real)
       AT( 6.30000E-01,  9.00000E-01) SIZE ( 2.12500E-01,  1.00000E-01);
      LiquidOut: (L: OUT real, x: OUT real)
```

```
    AT( 1.62500E-01,  0.00000E+00) SIZE ( 2.17500E-01,  1.00000E-01);
   VapourIn: (Vin: IN real, yin: IN real)
    AT( 6.30000E-01,  0.00000E+00) SIZE ( 2.20000E-01,  1.00000E-01);

  PAR
    L0 = 250.0; M0 = 20.0; alfa = 2.7; beta = 0.25;
  STATE
    m, mx:real;
  EQUATIONS
    m'DER = Lin - L;
    mx'DER = xin*Lin - x*L + yin*Vin - y*V;
    x = IF m > 0.0 THEN mx/m ELSE 0.0 END IF;
    y = alfa*x/(1.0 + (alfa - 1.0)*x);
    L = IF m > 0.0 THEN L0 + (m - M0)/beta ELSE 0.0 END IF;
    V = Vin;
  SECTION INIT;
    m = 20.0;
    mx= 18.0;
  END;

 Tray3: OBJECT
  AT( 1.52500E-01,  3.40000E-01) SIZE ( 2.85000E-01,  3.50000E-02);
  INTERFACE
   LiquidIn: (Lin: IN real, xin: IN real)
    AT( 1.60000E-01,  9.00000E-01) SIZE ( 2.15000E-01,  1.00000E-01);
   VapourOut: (V: OUT real, y: OUT real)
    AT( 6.30000E-01,  9.00000E-01) SIZE ( 2.12500E-01,  1.00000E-01);
   LiquidOut: (L: OUT real, x: OUT real)
    AT( 1.62500E-01,  0.00000E+00) SIZE ( 2.17500E-01,  1.00000E-01);
   VapourIn: (Vin: IN real, yin: IN real)
    AT( 6.30000E-01,  0.00000E+00) SIZE ( 2.20000E-01,  1.00000E-01);

  PAR
    L0 = 250.0; M0 = 20.0; alfa = 2.7; beta = 0.25;
  STATE
    m, mx:real;
  EQUATIONS
    m'DER = Lin - L;
    mx'DER = xin*Lin - x*L + yin*Vin - y*V;
    x = IF m > 0.0 THEN mx/m ELSE 0.0 END IF;
    y = alfa*x/(1.0 + (alfa - 1.0)*x);
    L = IF m > 0.0 THEN L0 + (m - M0)/beta ELSE 0.0 END IF;
    V = Vin;
  SECTION INIT;
    m = 20.0;
    mx= 18.0;
  END;

 Tray4: OBJECT
  AT( 1.47500E-01,  3.95000E-01) SIZE ( 2.82500E-01,  3.50000E-02);
  INTERFACE
```

```
  LiquidIn: (Lin: IN real, xin: IN real)
   AT( 1.60000E-01,  9.00000E-01) SIZE ( 2.15000E-01,  1.00000E-01);
  VapourOut: (V: OUT real, y: OUT real)
   AT( 6.30000E-01,  9.00000E-01) SIZE ( 2.12500E-01,  1.00000E-01);
  LiquidOut: (L: OUT real, x: OUT real)
   AT( 1.62500E-01,  0.00000E+00) SIZE ( 2.17500E-01,  1.00000E-01);
  VapourIn: (Vin: IN real, yin: IN real)
   AT( 6.30000E-01,  0.00000E+00) SIZE ( 2.20000E-01,  1.00000E-01);

 PAR
   L0 = 250.0; M0 = 20.0; alfa = 2.7; beta = 0.25;
 STATE
   m, mx:real;
 EQUATIONS
   m'DER = Lin - L;
   mx'DER = xin*Lin - x*L + yin*Vin - y*V;
   x = IF m > 0.0 THEN mx/m ELSE 0.0 END IF;
   y = alfa*x/(1.0 + (alfa - 1.0)*x);
   L = IF m > 0.0 THEN L0 + (m - M0)/beta ELSE 0.0 END IF;
   V = Vin;
 SECTION INIT;
   m = 20.0;
   mx= 18.0;
 END;

Tray5: OBJECT
 AT( 1.47500E-01,  4.45000E-01) SIZE ( 2.77500E-01,  4.00000E-02);
 INTERFACE
  LiquidIn: (Lin: IN real, xin: IN real)
   AT( 1.60000E-01,  9.00000E-01) SIZE ( 2.15000E-01,  1.00000E-01);
  VapourOut: (V: OUT real, y: OUT real)
   AT( 6.30000E-01,  9.00000E-01) SIZE ( 2.12500E-01,  1.00000E-01);
  LiquidOut: (L: OUT real, x: OUT real)
   AT( 1.62500E-01,  0.00000E+00) SIZE ( 2.17500E-01,  1.00000E-01);
  VapourIn: (Vin: IN real, yin: IN real)
   AT( 6.30000E-01,  0.00000E+00) SIZE ( 2.20000E-01,  1.00000E-01);

 PAR
   L0 = 250.0; M0 = 20.0; alfa = 2.7; beta = 0.25;
 STATE
   m, mx:real;
 EQUATIONS
   m'DER = Lin - L;
   mx'DER = xin*Lin - x*L + yin*Vin - y*V;
   x = IF m > 0.0 THEN mx/m ELSE 0.0 END IF;
   y = alfa*x/(1.0 + (alfa - 1.0)*x);
   L = IF m > 0.0 THEN L0 + (m - M0)/beta ELSE 0.0 END IF;
   V = Vin;
 SECTION INIT;
   m = 20.0;
   mx= 18.0;
```

```
 END;

FTray: OBJECT
 AT( 1.47500E-01,  5.00000E-01) SIZE ( 2.77500E-01,  3.25000E-02);
 INTERFACE
  LiquidIn: (Lin: IN real, xin: IN real)
   AT( 1.60000E-01,  9.00000E-01) SIZE ( 2.15000E-01,  1.00000E-01);
  VapourOut: (V: OUT real, y: OUT real)
   AT( 6.30000E-01,  9.00000E-01) SIZE ( 2.12500E-01,  1.00000E-01);
  LiquidOut: (L: OUT real, x: OUT real)
   AT( 1.62500E-01,  0.00000E+00) SIZE ( 2.17500E-01,  1.00000E-01);
  VapourIn: (Vin: IN real, yin: IN real)
   AT( 6.30000E-01,  0.00000E+00) SIZE ( 2.20000E-01,  1.00000E-01);

 PAR
   L0 = 350.0; M0 = 20.0; alfa = 2.7; beta = 0.25;
   Lfeed = 100.0; xfeed = 0.9;
 STATE
   m, mx:real;
 EQUATIONS
   m'DER = Lin - L + Lfeed;
   mx'DER = xin*Lin - x*L + yin*Vin - y*V + xfeed*Lfeed;
   x = IF m > 0.0 THEN mx/m ELSE 0.0 END IF;
   y = alfa*x/(1.0 + (alfa - 1.0)*x);
   L = IF m > 0.0 THEN L0 + (m - M0)/beta ELSE 0.0 END IF;
   V = Vin;
 SECTION INIT;
   m = 20.0;
   mx= 18.0;
 END;

Tray7: OBJECT
 AT( 1.47500E-01,  5.52500E-01) SIZE ( 2.80000E-01,  4.00000E-02);
 INTERFACE
  LiquidIn: (Lin: IN real, xin: IN real)
   AT( 1.60000E-01,  9.00000E-01) SIZE ( 2.15000E-01,  1.00000E-01);
  VapourOut: (V: OUT real, y: OUT real)
   AT( 6.30000E-01,  9.00000E-01) SIZE ( 2.12500E-01,  1.00000E-01);
  LiquidOut: (L: OUT real, x: OUT real)
   AT( 1.62500E-01,  0.00000E+00) SIZE ( 2.17500E-01,  1.00000E-01);
  VapourIn: (Vin: IN real, yin: IN real)
   AT( 6.30000E-01,  0.00000E+00) SIZE ( 2.20000E-01,  1.00000E-01);

 PAR
   L0 = 350.0; M0 = 20.0; alfa = 2.7; beta = 0.25;
 STATE
   m, mx:real;
 EQUATIONS
   m'DER = Lin - L;
   mx'DER = xin*Lin - x*L + yin*Vin - y*V;
   x = IF m > 0.0 THEN mx/m ELSE 0.0 END IF;
```

```
    y = alfa*x/(1.0 + (alfa - 1.0)*x);
    L = IF m > 0.0 THEN L0 + (m - M0)/beta ELSE 0.0 END IF;
    V = Vin;
  SECTION INIT;
    m = 20.0;
    mx= 18.0;
  END;

Tray8: OBJECT
  AT( 1.45000E-01,  6.10000E-01) SIZE ( 2.82500E-01,  4.25000E-02);
  INTERFACE
   LiquidIn: (Lin: IN real, xin: IN real)
    AT( 1.60000E-01,  9.00000E-01) SIZE ( 2.15000E-01,  1.00000E-01);
   VapourOut: (V: OUT real, y: OUT real)
    AT( 6.30000E-01,  9.00000E-01) SIZE ( 2.12500E-01,  1.00000E-01);
   LiquidOut: (L: OUT real, x: OUT real)
    AT( 1.62500E-01,  0.00000E+00) SIZE ( 2.17500E-01,  1.00000E-01);
   VapourIn: (Vin: IN real, yin: IN real)
    AT( 6.30000E-01,  0.00000E+00) SIZE ( 2.20000E-01,  1.00000E-01);

  PAR
    L0 = 350.0; M0 = 20.0; alfa = 2.7; beta = 0.25;
  STATE
    m, mx:real;
  EQUATIONS
    m'DER = Lin - L;
    mx'DER = xin*Lin - x*L + yin*Vin - y*V;
    x = IF m > 0.0 THEN mx/m ELSE 0.0 END IF;
    y = alfa*x/(1.0 + (alfa - 1.0)*x);
    L = IF m > 0.0 THEN L0 + (m - M0)/beta ELSE 0.0 END IF;
    V = Vin;
  SECTION INIT;
    m = 20.0;
    mx= 18.0;
  END;

Tray9: OBJECT
  AT( 1.40000E-01,  6.70000E-01) SIZE ( 2.82500E-01,  4.00000E-02);
  INTERFACE
   LiquidIn: (Lin: IN real, xin: IN real)
    AT( 1.60000E-01,  9.00000E-01) SIZE ( 2.15000E-01,  1.00000E-01);
   VapourOut: (V: OUT real, y: OUT real)
    AT( 6.30000E-01,  9.00000E-01) SIZE ( 2.12500E-01,  1.00000E-01);
   LiquidOut: (L: OUT real, x: OUT real)
    AT( 1.62500E-01,  0.00000E+00) SIZE ( 2.17500E-01,  1.00000E-01);
   VapourIn: (Vin: IN real, yin: IN real)
    AT( 6.30000E-01,  0.00000E+00) SIZE ( 2.20000E-01,  1.00000E-01);

  PAR
    L0 = 350.0; M0 = 20.0; alfa = 2.7; beta = 0.25;
  STATE
```

```
    m, mx:real;
  EQUATIONS
    m'DER = Lin - L;
    mx'DER = xin*Lin - x*L + yin*Vin - y*V;
    x = IF m > 0.0 THEN mx/m ELSE 0.0 END IF;
    y = alfa*x/(1.0 + (alfa - 1.0)*x);
    L = IF m > 0.0 THEN L0 + (m - M0)/beta ELSE 0.0 END IF;
    V = Vin;
  SECTION INIT;
    m = 20.0;
    mx= 18.0;
  END;

Boiler: OBJECT
  AT( 1.60000E-01,  1.35000E-01) SIZE ( 2.82500E-01,  8.50000E-02);
  INTERFACE
   LiquidIn: (Lin: IN real, xin: IN real)
    AT( 1.60000E-01,  9.00000E-01) SIZE ( 2.15000E-01,  1.00000E-01);
   VapourOut: (V: OUT real, y: OUT real)
    AT( 6.30000E-01,  9.00000E-01) SIZE ( 2.12500E-01,  1.00000E-01);
   IO: (Level: OUT real, bflow: IN real)
    AT( 9.00000E-01,  3.70000E-01) SIZE ( 1.00000E-01,  3.07500E-01);
   BotProd: (bf: OUT real, xb: OUT real)
    AT( 3.97500E-01,  0.00000E+00) SIZE ( 2.10000E-01,  1.00000E-01);

  PAR
    alfa = 2.7; area = 1.0;
  STATE
    m, mx:real;
  EQUATIONS
    m'DER = Lin - bf - V;
    mx'DER = xin*Lin - xb*bf - y*V;
    xb = IF m > 0.0 THEN mx/m ELSE 0.0 END IF;
    y = alfa*xb/(1.0 + (alfa - 1.0)*xb);
    bf = bflow;
    V  = 340.0;
    Level = m/area;
  SECTION INIT;
    m = 75.0;
    mx= 67.5;
  END;

Redrum: OBJECT
  AT( 4.50000E-01,  7.50000E-01) SIZE ( 3.02500E-01,  9.50000E-02);
  INTERFACE
   VapourIn: (Vin: IN real, yin: IN real)
    AT( 3.40000E-01,  9.00000E-01) SIZE ( 2.50000E-01,  1.00000E-01);
   IO: (Level: OUT real, dflow: IN real)
    AT( 9.00000E-01,  3.47500E-01) SIZE ( 1.00000E-01,  3.05000E-01);
   Reflux: (rf: OUT real, xd: OUT real)
    AT( 1.55000E-01,  0.00000E+00) SIZE ( 2.12500E-01,  1.00000E-01);
```

```
  TopProd: (df: OUT real, xd1: OUT real)
   AT( 6.47500E-01,  0.00000E+00) SIZE ( 2.05000E-01,  1.00000E-01);

 PAR
   area = 1.0;
 STATE
   m, mx:real;
 EQUATIONS
   m'DER = Vin - (rf + df);
   mx'DER = yin*Vin - xd*(rf + df);
   xd = IF m > 0.0 THEN mx/m ELSE 0.0 END IF;
   xd1= xd;
   df = dflow;
   rf = 250.0;
   Level = m/area;
 SECTION INIT;
   m = 500.0;
   mx= 450.0;
 END;

CONNECT
 Redrum.IO TO rdio
  VIA ( 7.52500E-01,  7.97500E-01) ( 8.22500E-01,  7.97500E-01)
      ( 8.22500E-01,  7.95000E-01) ( 8.22500E-01,  8.20000E-01)
      ( 9.00000E-01,  8.20000E-01) ;
 Tray9.LiquidOut TO Tray8.LiquidIn
  VIA ( 2.15000E-01,  6.70000E-01) ( 2.12500E-01,  6.52500E-01) ;
 Tray9.VapourIn TO Tray8.VapourOut
  VIA ( 3.45000E-01,  6.70000E-01) ( 3.47500E-01,  6.52500E-01) ;
 Tray8.LiquidOut TO Tray7.LiquidIn
  VIA ( 2.17500E-01,  6.10000E-01) ( 2.17500E-01,  5.92500E-01) ;
 Tray8.VapourIn TO Tray7.VapourOut
  VIA ( 3.50000E-01,  6.10000E-01) ( 3.50000E-01,  5.92500E-01) ;
 Tray7.LiquidOut TO FTray.LiquidIn
  VIA ( 2.25000E-01,  5.52500E-01) ( 2.25000E-01,  5.32500E-01) ;
 Tray7.VapourIn TO FTray.VapourOut
  VIA ( 3.50000E-01,  5.52500E-01) ( 3.50000E-01,  5.32500E-01) ;
 FTray.LiquidOut TO Tray5.LiquidIn
  VIA ( 2.22500E-01,  5.00000E-01) ( 2.22500E-01,  4.85000E-01) ;
 FTray.VapourIn TO Tray5.VapourOut
  VIA ( 3.50000E-01,  5.00000E-01) ( 3.50000E-01,  4.85000E-01) ;
 Tray5.LiquidOut TO Tray4.LiquidIn
  VIA ( 2.22500E-01,  4.45000E-01) ( 2.22500E-01,  4.30000E-01) ;
 Tray5.VapourIn TO Tray4.VapourOut
  VIA ( 3.47500E-01,  4.45000E-01) ( 3.50000E-01,  4.30000E-01) ;
 Tray4.LiquidOut TO Tray3.LiquidIn
  VIA ( 2.22500E-01,  3.95000E-01) ( 2.25000E-01,  3.75000E-01) ;
 Tray4.VapourIn TO Tray3.VapourOut
  VIA ( 3.55000E-01,  3.95000E-01) ( 3.55000E-01,  3.75000E-01) ;
 Tray2.LiquidIn TO Tray3.LiquidOut
  VIA ( 2.35000E-01,  3.25000E-01) ( 2.35000E-01,  3.40000E-01) ;
```

```
    Tray2.VapourOut TO Tray3.VapourIn
     VIA ( 3.62500E-01,  3.25000E-01) ( 3.62500E-01,  3.40000E-01) ;
    Tray1.LiquidIn TO Tray2.LiquidOut
     VIA ( 2.35000E-01,  2.80000E-01) ( 2.37500E-01,  2.95000E-01) ;
    Tray1.VapourOut TO Tray2.VapourIn
     VIA ( 3.62500E-01,  2.80000E-01) ( 3.62500E-01,  2.95000E-01) ;
    Tray1.LiquidOut TO Boiler.LiquidIn
     VIA ( 2.32500E-01,  2.50000E-01) ( 2.32500E-01,  2.20000E-01) ;
    Tray1.VapourIn TO Boiler.VapourOut
     VIA ( 3.67500E-01,  2.50000E-01) ( 3.67500E-01,  2.20000E-01) ;
    Boiler.IO TO bio
     VIA ( 4.42500E-01,  1.80000E-01) ( 6.60000E-01,  1.80000E-01)
         ( 6.60000E-01,  2.05000E-01) ( 9.00000E-01,  2.05000E-01) ;
    Tray9.VapourOut TO Redrum.VapourIn
     VIA ( 3.52500E-01,  7.10000E-01) ( 3.52500E-01,  8.62500E-01)
         ( 5.95000E-01,  8.62500E-01) ( 5.95000E-01,  8.45000E-01) ;
    Redrum.Reflux TO Tray9.LiquidIn
     VIA ( 5.30000E-01,  7.50000E-01) ( 5.30000E-01,  7.32500E-01)
         ( 2.12500E-01,  7.32500E-01) ( 2.12500E-01,  7.10000E-01) ;
  END;

  Reg1: CONTROLLER
   AT( 5.12500E-01, 6.45000E-01) SIZE ( 8.75000E-02, 1.42500E-01);
   INTERFACE
    io: (y: IN real, u: OUT real)
     AT( 0.00000E+00,3.45000E-01) SIZE ( 1.00000E-01,3.07500E-01);
    PAR
     K = -2.0; os = 90.0; yref = 500.0;
   EQUATIONS
     u = K*(yref - y) + os;
   END;

  Reg2: CONTROLLER
   AT( 5.07500E-01,  1.60000E-01) SIZE (8.75000E-02, 1.42500E-01);
   INTERFACE
    io: (y: IN real, u: OUT real)
     AT( 0.00000E+00,3.80000E-01) SIZE (1.00000E-01, 2.95000E-01);
     PAR
      K = -2.0; os = 10.0; yref = 75.0;
    EQUATIONS
      u = K*(yref - y) + os;
    END;

CONNECT
  Column.rdio TO Reg1.io
   VIA ( 4.10000E-01,  6.75000E-01) ( 4.60000E-01,  6.75000E-01)
       ( 4.60000E-01,  7.15000E-01) ( 5.12500E-01,  7.15000E-01) ;
  Column.bio TO Reg2.io
   VIA ( 4.10000E-01,  2.90000E-01) ( 4.55000E-01,  2.90000E-01)
       ( 4.55000E-01,  2.37500E-01) ( 5.07500E-01,  2.37500E-01) ;
END;
```