

EMSO: A New Environment for Modelling, Simulation and Optimisation

R. de P. Soares and A.R. Secchi*

Departamento de Engenharia Química - Universidade Federal do Rio Grande do Sul
Rua Sarmiento Leite 288/24 - CEP: 90050-170 - Porto Alegre, RS - Brasil

*Author to whom correspondence should be addressed. {rafael, arge}@enq.ufrgs.br

Abstract

A new tool, named EMSO (Environment for Modelling, Simulation and Optimisation), for modelling, simulation and optimisation of general process dynamic systems is presented. In this tool the consistency of measurement units, system solvability and initial conditions consistency are automatically checked. The solvability test is carried out by an index reduction method which reduces the index of the resulting system of differential-algebraic equations (DAE) to zero by adding new variables and equations when necessary. The index reduction requires time derivatives of the original equations that are provided by a built-in symbolic differentiation system. The partial derivatives required during the initialisation and integration are generated by a built-in automatic differentiation system. For the description of processes a new object-oriented modelling language was developed. The extensive usage of the object-oriented paradigm in the proposed tool leads to a system naturally CAPE-OPEN which combined with the automatic and symbolic differentiation and index reduction forms a software with several enhancements, when compared with the popular ones.

1. Introduction

Simulator is a valuable tool for applications ranging from project validation, plant control and operability to production increasing and costs reduction. This facts among others has made the industrial interest in softwares tools for modelling, simulation and optimisation to grow up, but this tools are still considered inadequate by the users (Che-Comp, 2002). The user dissatisfaction is mainly related with limited software flexibility, difficulty to use/learn and costly. Besides the lack of software compatibility and the slowness of inclusion of new methods and algorithms. Furthermore, the users have been pointed out some desired features for further development, like extensive standard features, *intelligent* interfaces among others (Hlupic, 1999).

In this work a new tool for modelling, simulation and optimisation of general dynamic systems, named EMSO (Environment for Modelling, Simulation and Optimisation) is presented. This tool aims to give the users more flexibility to use their available resources. The successful features found in the most used tools were gathered and some new methods were developed to supply missing features. In addition, some well established approaches from other areas were used, like the object-oriented paradigm. The big picture of the EMSO structure is shown at Figure 1 which demonstrates the modular architecture of the software.

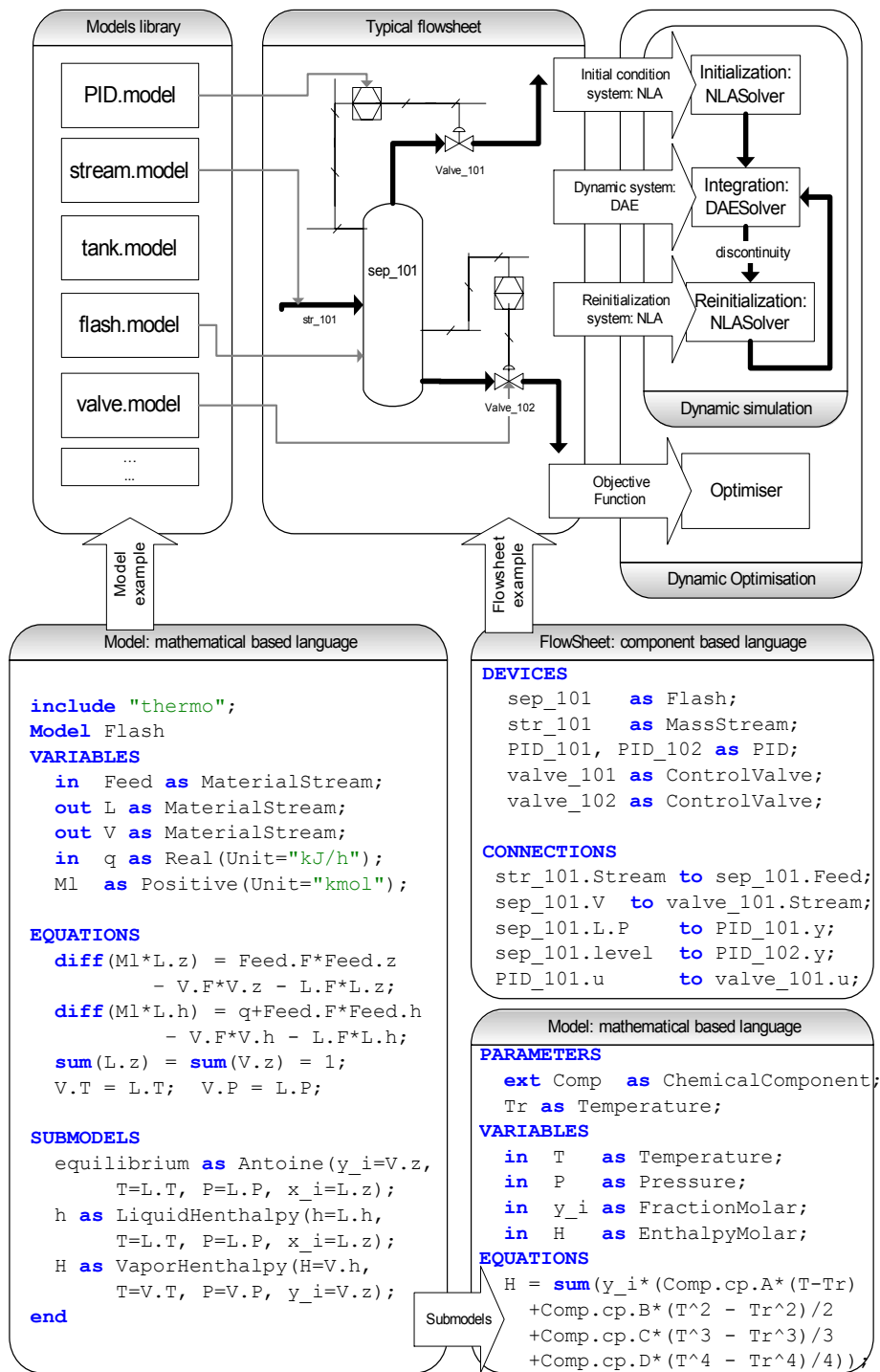


Figure 1. General vision of the EMSO structure and its components.

2. Process Model Description

In the proposed modelling language there are three major entities: *models*, *devices*, and *flowsheets*. *Models* are the mathematical description of some *device*; a *device* is an instance of a *model*; and a *flowsheet* represents the process to be analysed which is composed by a set of *devices*. At bottom of Figure 1 are given some pieces of code which exemplifies the usage of the language.

EMSO makes intensive use of automatic code generators and the object-oriented paradigm whenever is possible, aiming to enhance analyst and productivity.

2.1 Model

In the EMSO language, one *model* consists in the mathematical abstraction of some real equipment, process piece or even software. Examples of *models* are the mathematical description of a tank, pipe or a PID controller.

Each *model* can have parameters, variables, equations, initial conditions, boundary conditions and submodels that can have submodels themselves. *Models* can be based in pre-existing ones, and extra-functionality (new parameters, variables, equations, etc.) can be added. So, *composition* (*hierarchical modelling*) and *inheritance* are supported.

Every parameter and variable in a *model* is based in a predefined *type* and have a set of properties like a *brief* description, *lower* and *upper* bounds, *unit* of measurement among others. As *models*, *types* can have subtypes and the object-oriented paradigm is implemented. Some examples of *types* declarations can be seen in Figure 2.

```
Fraction as Real (Lower=0, Upper=1);
Positive as Real (Lower=0, Upper=inf);
EnergyHoldup as Positive (Unit="J");
ChemicalComponent as structure (
  Mw as Real (Unit="g/mol");
  Tc as Temperature (Brief="Critical Temperature");
  Pc as Pressure;
);
```

Figure 2. Examples of type declarations.

2.2 The Flowsheet and its Devices

In the proposed language a *device* is an instance of a *model* and represents some *real* device of the process in analysis. So, a unique *model* can be used to represent several different *devices* which have the same structure but may have different conditions (different parameters values and specifications). *Devices* can be connected each other to form a *flowsheet* (see Figure 1) which is an abstraction for the real process in analysis.

Although the language for description of *flowsheets* is textual (bottom right in Figure 1), it is simple enough to be entirely manipulated by a graphical interface. In which *flowsheets* could be easily built by dragging *model* objects into it to create new *devices* that could be connected to other *devices* with the aid of some pointing unit (mouse).

3. Consistency Analysis

In solving the resulting system of differential-algebraic equations (DAE) of a *flowsheet*, prior analysis can reveal the major failure causes.

There are several kinds of consistency analysis which can be applied in the DAE system coming from the mathematical description of a dynamic process. Some of them are: measurement units, structural solvability and initial conditions consistency.

3.1 Measurement Units Consistency

In modelling physical processes the conversion of measurement units of parameters is a tiresome task and prone to error. Moreover, a ill-composed equation usually leads to a measurement unit inconsistency. For this reasons, in EMSO the measurement units consistency and units conversions are automatically made for all equations, parameter setting and connections between *devices*.

Once all expressions are internally stored in a symbolical fashion and all variables and parameters holds its measurement units, the units consistency can be easily tested with the aid of the units measurement handling package RUnits (Soares, 2002).

3.2 DAE Solvability

Soares and Secchi (2002) have proposed a structural method for index reduction and solvability test of DAE systems. With this method, structural singularity can be tested and the structural differential index can be reduced to zero by adding new variables and equations. Such variables and equations are the derivatives of the original ones with respect to the independent variable.

EMSO makes use of this method, allowing the solution of high-index DAE problems without user interaction. The required derivatives of the variables and equations are provided by a built-in symbolic differentiating system.

3.3 Initial Conditions Consistency

Once a DAE system is reduced to index zero the dynamic freedom degree is determined. So, the initial condition consistency can be easily tested by an association problem as described by Soares and Secchi (2002). This approach is more robust when compared with the index-one reduction technique presented by Costa et al. (2001).

4. External Interfaces

Usually each simulation software vendor has its proprietary interfacing system, this leads to heterogeneous systems. Recently, the CAPE-OPEN project (CO-LAN, 2002) has published open standards interfaces for computer-aided process engineering (CAPE) aiming to solve this problem. EMSO complies with this open pattern. The interfaces are implemented natively rather than *wrapping* some other proprietary interface mechanism, and CORBA (OMG, 1999) was used as the middleware.

The extensive usage of the interfaces turns its efficiency a priority. For this reason some modifications for the numerical CAPE-OPEN package (CO-LAN, 2002) were proposed. This modifications consists in changing some function calling conventions, more details can be seen in Soares and Secchi (2002b).

5. Graphical User Interface

The graphical user interface (GUI) of EMSO combines *model* development, *flowsheet* building, process simulation and results visualising and handling all in one. EMSO is entirely written in C++ and is designed to be very modular and portable. In running tasks there are no prior generation of intermediary files or compilation step, everything is made out at the memory. The software is multithread, allowing real-time simulations and even to run more than one *flowsheet* concurrently without blocking the GUI. Furthermore, calculations can be paused or stopped at any time. The Figure 3 shows the EMSO GUI, it implements a Multiple Document Interface (MDI).

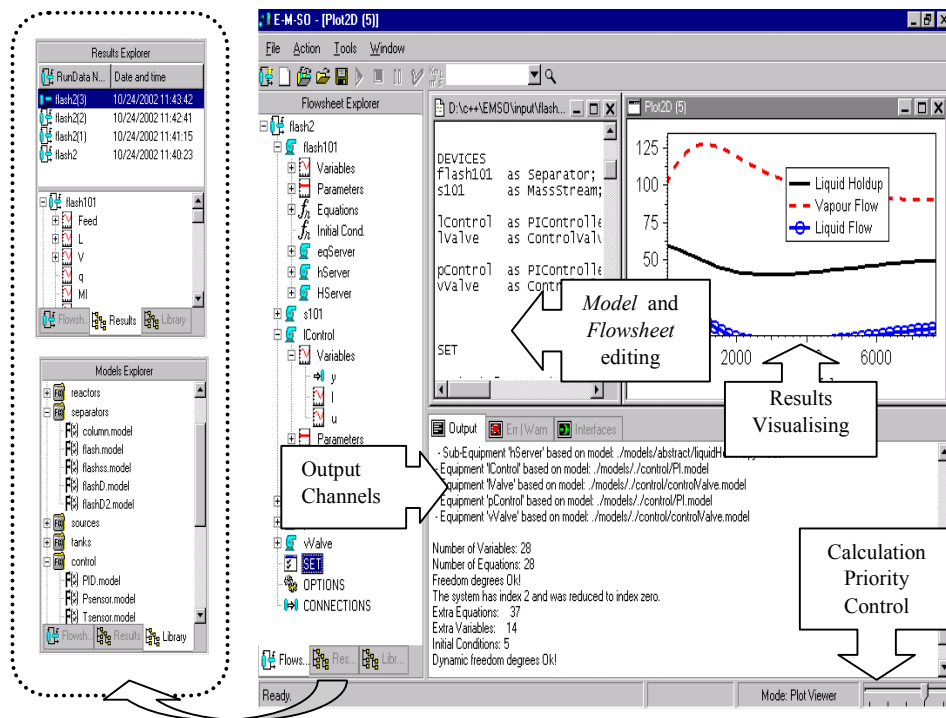


Figure 3. EMSO graphical user interface.

6. Applications

Consider the separation process outlined at *Figure 1*. It consists in a flash vessel with level and pressure control and was modelled as:

$$\frac{d}{dt}(Ml.x_i) = F.z_i - L.x_i - V.y_i \quad \frac{d}{dt}(Ml.h) = q + F.h_F - L.h - V.H \quad (1a)$$

$$\sum_i x_i = \sum_i y_i = 1 \quad y_i = K_i \cdot x_i \quad (1b)$$

$$K_i = f_1(T, P, x_i, y_i) \quad (2)$$

$$h_F = f_2(T_F, P_F, z_i), \quad h = f_3(T, P, x_i), \quad H = f_4(T, P, y_i) \quad (3)$$

$$V = f_5(P), \quad L = f_6(Ml) \quad (4)$$

where F , V and L are the feed, vapour and liquid molar flow rates, h_F , H and h are the respective molar enthalpies, T is the temperature and P pressure, z_i , x_i and y_i are the feed, liquid and vapour molar fractions and q is the heat duty.

In EMSO this process can be modelled in a modular fashion: Eq.(1) as *flash device*; Eq.(2) as *thermodynamic equilibrium device*; Eq.(3) as *enthalpy devices*; Eq.(4) as *control devices*.

The dynamic simulation of this system is trivial if q and the feed conditions are known along the time and T , Ml and x_i are given as initial conditions. But if a temperature profile is specified instead of q a high-index system take place and cannot be directly solved by the popular simulation tools.

In solving this problem EMSO reports a index two system and automatically reduces the index to zero. In the Figure 4 the composition profiles of the outlet streams are showed for a start-up condition of this index two system.

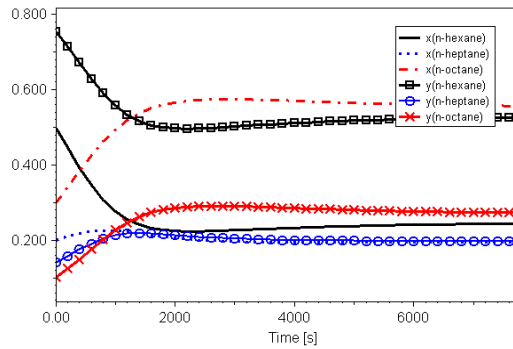


Figure 4. Solution of the index two separation process.

The index-three batch distillation column proposed by Logsdson and Biegler (1993) was also successfully solve by EMSO, but there was no room to show the results.

7. Conclusions

An object-oriented language for modelling general dynamic process was successfully developed and its usage has proved efficiency in code reusability. The development of model libraries of models for thermodynamics, process engineering and other application areas is one of the future tasks. The DAE index reduction method allows

EMSO to directly solve high-index DAE systems without user interaction. This fact combined with the symbolic and automatic differentiation systems and the CAPE-OPEN interfaces leads to a software with several enhancements.

Dynamic optimisation is at design stage, but the modular internal architecture of EMSO allows it to be added further without re-coding the other modules.

8. References

- Che-Comp, 2002, The State of Chemical Engineering Softwares, www.che-comp.org .
- CO-LAN, 2002, Conceptual Design Document for CAPE-Open Project, www.co-lan.org .
- Costa Jr., E.F., R.C. Vieira, A.R. Secchi and E.C. Biscaia, 2001, Automatic Structural Characterization of DAE Systems, ESCAPE 11, Kolding, Denmark, 123-128.
- Hlupic, V., 1999, Simulation Software: User's Requirements, Comp. Ind. Engrg, 37, 185-188.
- Logsdon, J. S. and Biegler, L. T., 1993, Ind. Eng. Chem. Res., v.32, n.4, 692-700.
- Luca, L. De and Musmanno, R., 1997, A Parallel Automatic Differentiation Algorithm for Simulation Models, Simulation Practice and Theory, 5, 235-252.
- OMG, 1999, The Common Object Request Broker, version 2.3.1, www.omg.org.
- Soares, R. de P. and Secchi, A. R., 2002, Direct Initialization and Solution of High-Index DAE Systems with Low-Index DAE solvers, Comp. Chem. Engrg (submitted 2002).
- Soares, R. de P. and Secchi, A. R., Efficiency of the CAPE-OPEN Numerical Open Interfaces, Technical Reporting, UFRGS, Porto Alegre, Brasil (2002b).
- Soares, R. de P., Runits: the Measurement Units Handling Tool - Reference Manual, rafael@enq.ufrgs.br (2002).