NMF-BASED ASPECT MODELS IN STEP/EXPRESS FOR BUILDING AND PROCESS PLANT SIMULATION

Per Sahlin
Building Services Engineering
Royal Institute of Technology
100 44 STOCKHOLM, SWEDEN
e-mail: plurre@kth.se

Curt Johansson
Construction Management
Royal Xnstitute of Technology
100 44 STOCKHOLM, SWEDEN
e-mail: curt@ce.kth.se

Abstract

Automated design performance assessment through simulation will be an important aspect of future product model technology. The research in this area has so far been focused on traditional simulation tools. However, the rapid development of new structurally different tools calls for a shift of attention. New object-oriented methods of describing simulation models can and should be integrated with the product model itself. In this paper we will briefly review a current development trend in continuous simulation and present a new language for model description, Neutral Model Format (NMF), which in recent years has gained considerable attention in the field of building simulation. The possibility of joining the continued NMF development with the STEP domain is discussed and some examples of NMF based EXPRESS models are presented.

1. INTRODUCTION

One driving factor behind product model research is that it **will** give designers direct access to easy and repeated design evaluation. Obviously, cost estimates, bills of materials, and various drawings should be easily generated from product model data, but of equal importance are measures of the dynamical performance of the design at hand. In the AEC field the EEC COMBINE project (phase 1) has demonstrated feasibility of data mapping from an EXPRESS-based data model of a building **to** a range of established building performance evaluation **(BPE)** tools [Augenbroe 19931. Phase 2 of this project seeks to put this technology to use among practitioners in the field. Another industrial sector with considerable activity in both product modelling and simulation is the process industry.

The authors of this paper have for some time worked with new simulation techniques and languages for continuous modular systems. These techniques are applicable to a large class of static and dynamical simulation problems in, e.g., the building, energy and process industries. One important aspect of this work has been involvement in the definition of a standard format, Neutral Model Format (NMF), for expression of component level simulation models. The purpose of this paper is to investigate the applicability of STEP technology in the continuation of this work.

Currently, *component models* (primitive models) are automatically translated from NMF to the proprietary format of the target simulation environment. For example, an NMF model of an axial fan is used to generate an axial fan class in, e.g. **IDA** [Sahlin 19911. The class is then instantiated in the target environment. The instances are furnished with suitable parameters, and incorporated into a system model. The next natural step in the NMF development is to formulate an environment independent way of expressing and communicating instantiated *system models* as well. Several authors have already suggested and even implemented such NMF extensions [Kolsaker 1994a, Lorenz 1994]. Since object oriented simulation is a highly relevant topic for product modelling efforts [Augenbroe 19911, we will analyze the implications of using EXPRESS for data modelling of NMF instantiated system models.

In the next two sections a brief overview is given of current work on so called object oriented simulation methods, mainly in the context of building simulation, and of the Neutral Model Format.

2. OBJECT ORIENTED SIMULATION ENVIRONMENTS

The term object oriented is perhaps not the best descriptor for these tools but it has nevertheless become widely used and we will use it here as well. The object orientation concerns mainly the modularity of the physical systems that are being modelled and not so much software techniques. Naturally, most recent developments also use object oriented programming to varying degrees.

2.1 PHYSICAL SYSTEMS AND MATHEMATICAL MODELS

Physical systems that we aim to simulate are modular in nature, i.e. they naturally decompose into subsystems. Frequently, identical subsystems are repeated a number of times in a model, a fact that is taken advantage of in many tools. Furthermore, the systems should have a basically continuous behavior, meaning that equations used to describe them, as well as forcing functions, will have a limited number of discontinuities. Purely event driven systems are excluded.

Models may be expressed in several ways. Bond graphs, linear graphs, block diagrams, electrical analogies, and mathematical equations are frequently used modes of expression. Also used, for mainly historical reasons, are subroutines in some programming language. A discussion of pros and cons of various methods of description can be found in [Lorenz 19871.

If characterized by equations, the physical systems under consideration will require both algebraic and differential equations. Differential equations can be either ordinary (ODE) or partial (PDE), although current tools require that PDEs are explicitly discretized in space and thus turned into ODEs. Note that in contrast to many widely used commercial tools the simulation environments we are concerned with here are not limited to ODEs only. They allow a free mixture of algebraic and ordinary differential equations generally referred to as differential-algebraic systems of equations (DAE).

Furthermore, the simulation tools under discussion are rarely used for applications where a strict formalism for generating governing equations exists. In, e.g., electrical circuit analysis, multibody mechanics, or structural analysis special purpose systems may be more advantageous.

Examples of physical systems that fit this description can be found in many fields. Chemical process plant simulation is a significant area of application. Energy distribution networks and plants is another. The authors of this paper have mainly worked with building related systems and important applications within this field are: thermal processes in walls and spaces; air and water based distribution systems and plants; and automatic control.

2.2 SEPARATION OF MODELLING AND SOLVING ACTIVITIES

In contrast to many established design tools, e.g. in building simulation, OOSEs separate strictly between the modelling and subsequent system solution activities. A modelling tool is often used for model formulation. This tool generates a system model, generally expressed in a *modelling language*. The model is then treated by a solver. An important benefit of a separate solver is that it may be altered or even exchanged with minimal interference with the modelling environment.

Key characteristics of the modelling language, such as expressiveness and level of standardization, are critical to the usefulness and development potential of the overall OOSE. The Neutral Model Format is part of such a modelling language. This paper describes one way towards a complete modelling language that may be standardized.

2.3 TARGET USERS AND SOFTWARE STRUCTURE

Most of the simulation tools under discussion are intended for quite sophisticated users, who are well versed in mathematical modelling, numerical methods and advanced use of computers. These tools are not directly suited for designers, without special simulation expertise, that use simulation as one of several methods for design evaluation. However, for the expert, they generally provide an efficient graphical environment for model building, simulation and analysis.

Other tools, e.g. EKS and IDA, are primarily intended for efficient design tool production, and the normal end user will rarely interact directly with the underlying OOSE techniques.

2.4 AVAILABLE AND EMERGING OOSEs

A few tools and environments with the discussed main characteristics are already matured and available and others are under development. Among the available ones are e.g.:

TRNSYS was developed during the seventies at the Solar Energy Lab at the University of Wisconsin. It was one of the first modular simulation solvers for DAEs and it is distributed as a Public Domain product. Several compatible modelling tools have been developed, e.g. PRESIM.

HVACSIM+ is a solver with similar characteristics as TRNSYS in terms of model format and structure, but more recent numerical techniques are utilized. It was developed by NIST in Maryland and released in the mid eighties on a Public Domain basis.

SANDYS is a general DAE solver and textual modelling environment developed by ASEA, Sweden, in the early eighties. It is commercially available from ABB Corporate Research.

ALLAN-NEPTUNIX is a graphical modeller and solver combination developed by Gaz de France and CISI Engineering. It is since a few years commercially available from the developers.

ESACAP is a recently developed DAE solver by the European Space Agency. It is commercially available from STANSIM, Denmark.

DYMOLA is a text based commercial modelling tool with symbolic algebra capabilities and interfaces to several solvers. A GUI is under development. Available from DYNASIM, Lund, Sweden.

Some tools under development are:

CLIM 2000, a graphical modelling tool for building applications, is developed by Electricite de France.

MS1 is a graphical multi input language modeller with interfaces to several solvers by Lorenz Consulting, Liege, Belgium in cooperation with Electricite de France.

IDA, a graphical modelling environment and solver, is under development at the Swedish Institute of Applied Mathematics.

SPARK is a solver and graphical model editor under development at LBL, Berkeley, California.

OMSIM is a graphical modelling tool under development at the Dept. of Automatic Control at the Lund Institute of Technology, Sweden.

EKS is a C++ toolkit for development of energy related simulation design tools, by among others the Univ. of Strathclyde, Scotland.

3. THE NEUTRAL MODEL FORMAT

Without a comprehensive, validated library of ready made component models in a relevant application area most simulation environments are rather useless. To develop all necessary models from scratch is, in most projects, quite unrealistic. And since the cost of developing a substantial library easily exceeds the development cost of the simulation tool itself, it is important to be able to reuse what other people already have done. This was the basic motivation for proposing a text based neutral model format to the building simulation community in 1989 [Sahlin and Sowell 1989]. Since then the proposal has attracted a great deal of interest from environment developers and users in several application fields. Prototype translators have been developed for IDA [Kolsaker 1994a], SPARK [Nataf 1994] and ESACAP [Pelletret 1994a]. Translator development projects have been funded for TRNSYS, HVACSIM+ [ASHRAE 1994], and MS1 [Lorenz 1994]. Export and import capabilities are planned and partly implemented for ALLAN-NEPTUNIX [Jeandel 1994].

Pending formal standardization, ASHRAE (American Society of Heating, Refrigerating, and Air-conditioning Engineers) has formed an ad hoc committee that approves changes to the present format.

NMF has two main objectives: (1) models can be automatically translated into the local representation of several simulation environments, i.e. the format is program *neutral* and machine readable; and (2) models should be easy to understand and express for non-experts. The first objective enables development of common model libraries, which can be accessed from a number of simulation environments.

3.1 BASIC NMF FEATURES

Internal component model behavior is described by a combination of algebraic and ordinary differential equations. Equations may be written in any order and in the form

<expression> = <expression>;

NMF only *states* equation models, while *solution* of equations is, in some cases, left to the target environment (e.g. IDA, or SPARK), or the NMF translator in others (e.g. TRNSYS, or HVACSIM+).

NMF supports model encapsulation through a link concept, i.e. models may only interact via variables appearing in LINK statements. To enhance and encourage model plug compatibility, links and variables are globally typed. The idea is that basic list of such types should be included in each revision of the standard, but that users may add to the list as need arise. A selection of such global types is:

```
QUANTITY_TYPES

/* type name       unit              kind */

   Area            "m2"              CROSS
   Control         "dimless"         CROSS
   Density         "kg/m3"           CROSS
   Factor          "dimless"         CROSS
   HeatCap         "J/(K)"           CROSS
   HeatCapA        "J/(K m2)"        CROSS
   HeatCapM        "J/(kg K)"        CROSS
   HeatCond        "W/(K)"           THRU
   HeatFlux        "W"               THRU
   HeatFlux_k      "kW"              THRU
   Temp            "Deg-C"           CROSS

LINK_TYPES

/* type name        variable types... */

/* generic         (arbitrary, arbitrary,...) implicitly
defined */
   F               (Force)
   FL              (Force,Length)
   Q               (HeatFlux)
   T               (Temp)
   PMT             (Pressure, MassFlow, Temp)
   PMTQ            (Pressure, MassFlow, Temp, HeatFlux)

   MoistAir        (Pressure, MassFlow, Temp, HumRatio)
   BidirFlow       (Pressure, MassFlow, Enthalpy, HeatFlux)
```

A quantity type includes a physical unit and information about potential (across) or flow (through) type. A link type is simply an ordered list of quantity types. Let us now look at an example of a rather simple NMF model using the heat equation in one dimension.

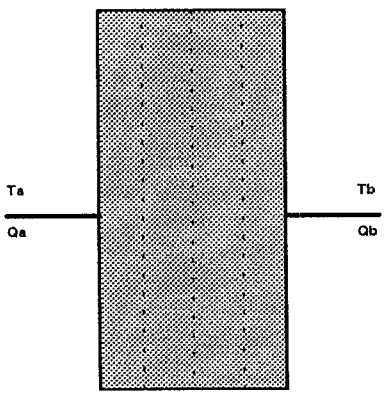


Figure 1. A finite difference model of a wall with one homogeneous layer. Temperature and heatflux on each terminal.

```
CONTINUOUS_MODEL tq_hom_wall

ABSTRACT
"A 1D finite difference wall model. One homogeneous layer.
TQ interfaces on both sides."

EQUATIONS
/* space discretized heat equation */
c_coeff * T'[1] = Taa - 2.*T[1] + T[2] ;
c_coeff * T'[n] = T[n - 1] - 2. * T[n] + Tbb ;

FOR i = 2, (n -1)
c_coeff * T'[i] = T[i - 1] - 2. * T[i] + T[i + 1];
END_FOR ;
/* boundary equations */
0 = -Ta + .5 * (Taa + T[1]) ;
0 = -Tb + .5 * (T[n] + Tbb) ;
0 = -Qa + d_coeff * (Taa - T[1]) ;
0 = -Qb + d_coeff * (Tbb - T[n]) ;

LINKS
/*  type            name       variables ....  */
    TQ              a_side     Ta, POS_IN Qa ;
    TQ              b_side     Tb, POS_IN Qb ;

VARIABLES
/* type     name         role def     min     max      description*/
  Temp      T[n]         OUT   20. abs_zero  BIG  "temperature profile"
  Temp      Ta           OUT   20. abs_zero  BIG  "a-side surface temp"
  Temp      Tb           OUT   20. abs_zero  BIG  "b-side surface temp"
  Temp      Taa          OUT   20. abs_zero  BIG  "a-side virtual temp"
  Temp      Tbb          OUT   20. abs_zero  BIG  "b-side virtual temp"
  HeatFlux  Qa           IN     0.    -BIG   BIG  "a-side entering heat"
  HeatFlux  Qb           IN     0.    -BIG   BIG  "b-side entering heat"

MODEL_PARAMETERS
/* type     name        role  def mi  max        description  */
  INT       n           SMP    3   3  BIGINT "number of temp layers"

PARAMETERS
/*   type     name        role    [def    [min    max]]   description*/

/* supplied parameters */
  Area       a             S_P    10.  SMALL BIG  "wall area"
  Length     thick         S_P   .2    SMALL BIG  "wall total thickness"
  HeatCondL  lambda        S_P   0.5   SMALL BIG  "heat transfer coeff"
  Density    rho           S_P   2000  SMALL BIG  "wall density"
  HeatCapM   cp            S_P   900.  SMALL BIG  "wall heat capacity"

/* computed parameters */
  generic    d_coeff       C_P              "lambda*a/dx"
  Length     dx            C_P              "layer thickness"
  generic    c_coeff       C_P              "rho*cp*dx*dx/(lambda*3600.)"


PARAMETER_PROCESSING
dx := thick / n ;
c_coeff := rho * cp * dx * dx / (lambda * 3600.) ;
d_coeff := lambda * a * dx ;

END_MODEL
```

To enable direct model translation to input-output oriented environments (e.g. TRNSYS, or HVACSIM+), variable declarations have a role attribute indicating IN for given variables and OUT for calculated ones.

Variables and parameters may be vectors or matrices. A parameter is anything that must remain constant throughout every simulation. Links may also be vectors, thus allowing models with variable number of ports. Vector and matrix dimensions are governed by a special type of parameter, model parameters. Regular and model parameters are divided into two categories, user supplied and computed, algorithmic computation of which is described in the parameter processing section.

Arbitrary foreign functions in Fortran 77 or C may be defined, either globally or locally within a model.

Special functions are defined to handle discontinuities, hysteresis, linearization, and errors. A more complete account of NMF is given in the reference report [Sahlin, Bring, and Sowell 1994].

3.2 NMF DEVELOPMENT DIRECTIONS

Currently, a reasonable agreement about the NMF grammar has been reached. Developers can count on stability of the present format and backward compatibility. This enables us to get on with the work of defining NMF-based component model libraries and to develop further NMF translators. Several substantial model libraries have already been developed and many more are underway.

Regarding the format itself, several extensions have been suggested. In the discussion of these it is important to bear in mind that, at the time of the original proposal, NMF was not primarily intended as a replacement of existing proprietary model languages, but as a complement, enabling component model exchange and library building.

Planned extensions and supporting tools that fall within the scope of the original NMF intentions are:

1. An NMF handbook with style guidelines for model architecture. The current NMF manual is completely insufficient as a pedagogical tool. (Encompassed by funded project [ASHRAE 1994].)

2. Model documentation guidelines and templates, storage and retrieval mechanisms. This area is addressed by Pelletret in a recent (draft) proposal [Pelletret 1994]. The ESPRIT OLMECO project - development of a large mechatronics library - is another source of inspiration.

3. Investigation regarding adaptable models, through property inheritance and/or through hierarchical modelling. Property inheritance between models may result in better model reuse but it will on the other hand also have negative effects on model

portability, since inheritance trees must be passed when shipping a model. This leads to reconciliation problems if a similar, but not identical, tree exists on the receiving side.

4. Model library structure and management tools, including mechanisms for model browsing and retrieval.

5. Discrete time (sampling) models. This is necessary to study sampling control circuits.

There are several additional items that belong in this list - such as formal rules for permitted model connections and a language or keyword system for expression of model assumptions - that are omitted here due to space.

In the context of a complete modelling language the present format lacks the ability to express:

1. Component model instances, with parameter values, initial values of all variables, and information about boundary variables..

2. Hierarchical systems of such instances.

3. Numerical simulation parameters, such as tolerances, stepsize limits, algorithm selection commands, that can be generalized for a large class of solvers.

4. Graphical schemata for user presentation of simulation models. Large models are much easier to comprehend if they are described graphically.

The drive for development of a complete NMF-based modelling language comes primarily from developers of new modelling tools, who see little reason to develop proprietary formats. Two such developers have made concrete proposals and implementations are well underway [Lorenz 1990], [Kolsaker 1994].

4. WHY STEP/EXPRESS?

STEP (STandard for the Exchange of Product model data) is an international standard for product descriptions [ISO TC 184 1993]. The data for these descriptions are modelled in a special language called EXPRESS, which is in itself part of the STEP standard. EXPRESS is an object oriented language that is particularly well suited for information modelling. A subset of EXPRESS is EXPRESS-G, a fully graphical language for data modelling. EXPRESS-G schemata can automatically be translated into textual EXPRESS code, which in turn can be translated into, e.g., C++ class definitions. A number of tools and related standards are (and will be) available for STEP/EXPRESS. A (default) textual representation of any EXPRESS schema is for example implicitly defined (STEP physical file).

Since the first proposal in 1989 the discussion about various NMF-constructs has focused on the grammar. The textual appearance of selected models has been the main object. This is of course quite appropriate for the equation core of component

models, but for instantiated system models and related data it may be more fruitful to regard data models directly, and to treat textual representation as one of several possible views. EXPRESS seems to be an appropriate vehicle for the future NMF discussion. Further reasons for the employment of STEP technology include:

- Simulation models will most likely be an important aspect of many product model applications, and they should therefore be encompassed by STEP, either as pure aspect models or as parts of global models

- Many existing STEP/EXPRESS resources will be useful for development of NMF-oriented application tools

- The fact that STEP physical files most likely will be more difficult to read (for humans) than a tailored high level language is of little consequence for realistic-size simulation models, which generally are of such magnitude that they rarely are printed and studied in their raw form

4.1 PRESENT NMF IN STEP

In this our initial work we have chosen to focus directly on the imminent problem of defining conceptual models of NMF instances, and of hierarchical systems of such instances. This means that nothing is said about the internal behavior, e.g. equations, of a model. Only its state is encompassed, and it is assumed that the underlying NMF model is known to all parties.

Another interesting issue is of course the conceptual models of internal behavior as well, i.e. to model the present NMF in EXPRESS, with entities such as equation, if_then_else_clause, etc. Such models are necessary for development of NMF parsers and translators.

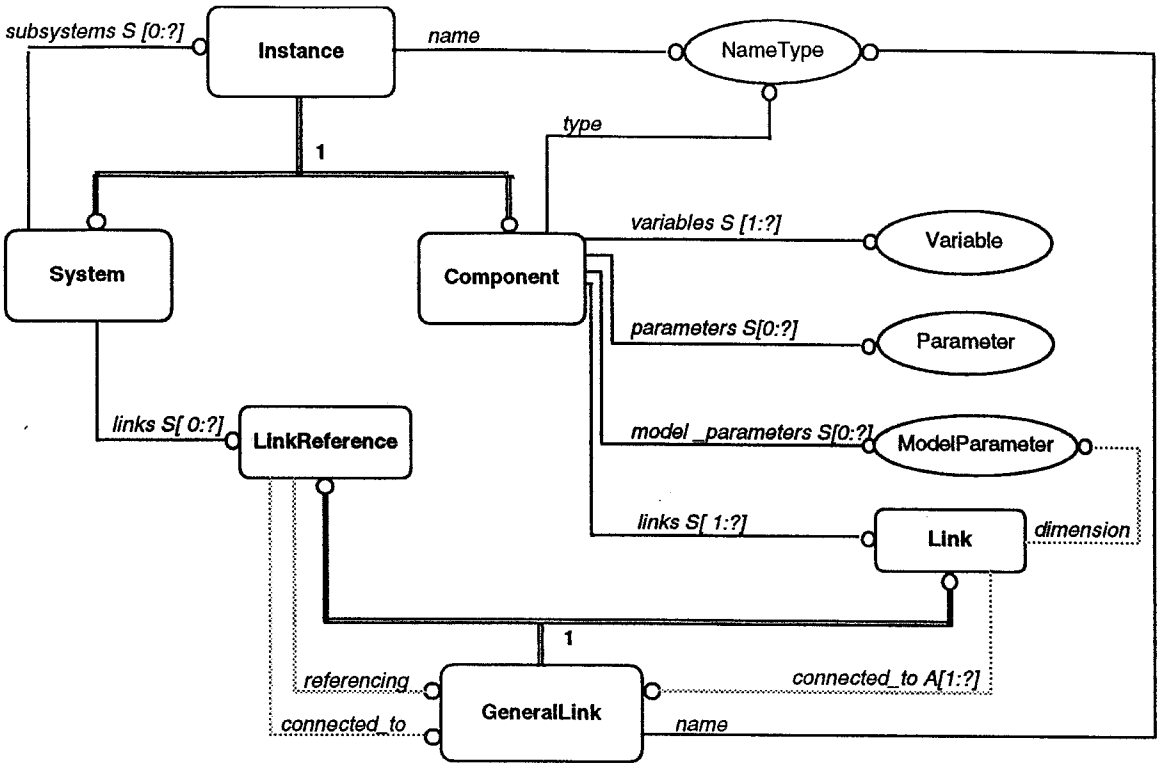
The main motivation for remodelling the present NMF in EXPRESS is completeness. New component models could be communicated with the same tools and protocols. A potential EXPRESS-based STEP standard would not have to rely on an additional non-EXPRESS standard.

Additional benefits can be expected for design and implementation of NMF component model databases and management tools.

The present conclusion is that it would be worthwhile to model the present NMF in EXPRESS. However, since the discussion of instances and systems can be carried out separately, we have chosen to focus on this in our initial work.

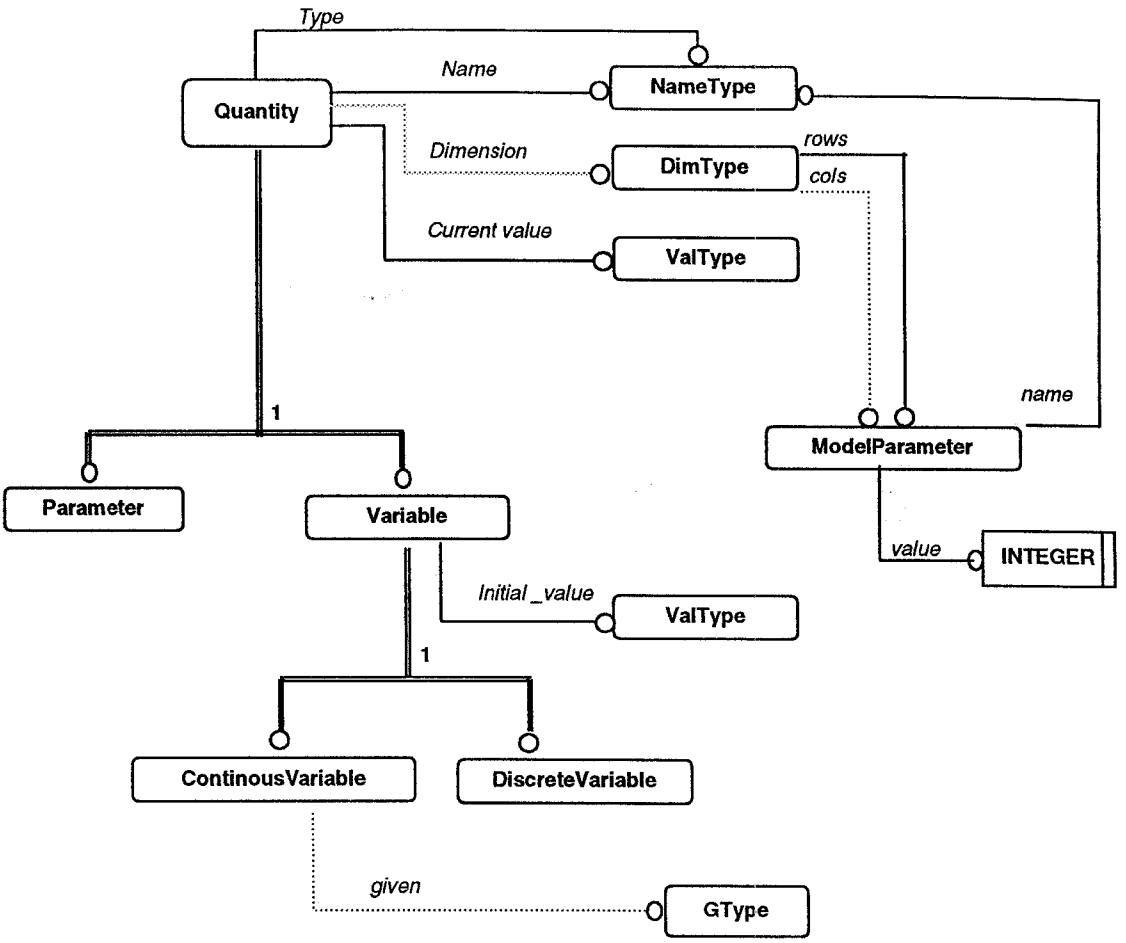
5. NMF MODEL INSTANCES IN EXPRESS-G

In the following an EXPRESS-G representation of NMF component and system model instances is presented. An instance is a specific occurrence of a model expressing the full state, in terms of its parameter values, variable values, and associated data. Schemata 1 through 4 shows the EXPRESS-G representation of this data.



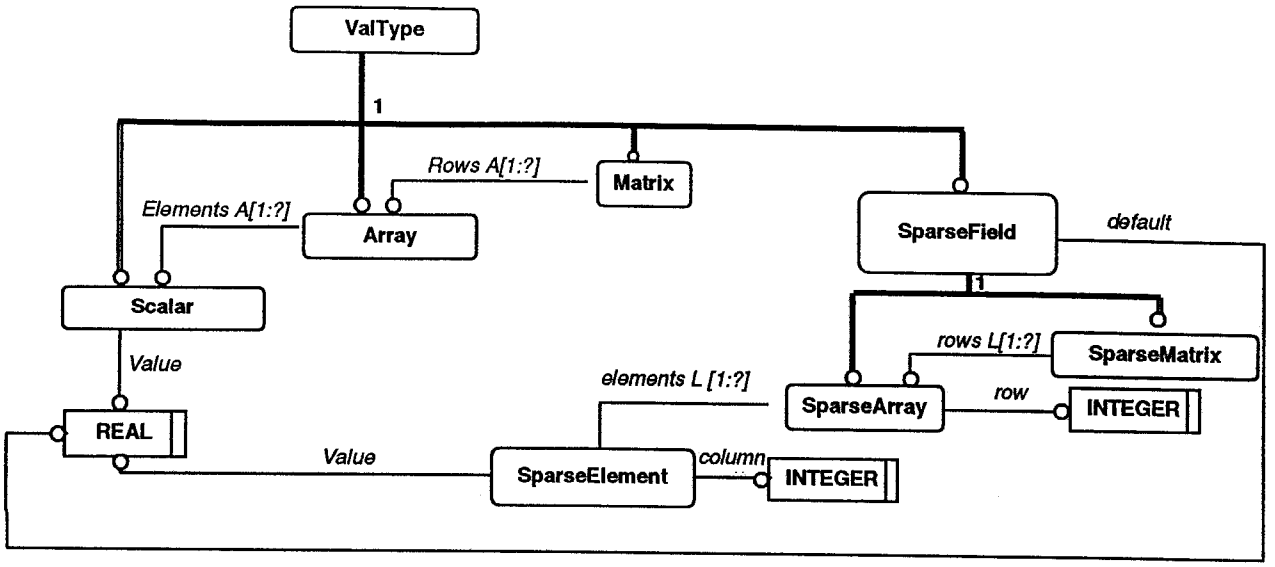
Schema 1

Schema 1 shows the structure of an NMF model instance, which may appear as either a System, with references to underlying subsystems, or as a Component with object bags for variables, parameters, model parameters, and links, each of which is specified more closely in the following schemata . Model parameters are named integers that are used for dimensioning of arrays and matrices. Links are the connection ports of Components. The ports of a System are called LinkReferences. They provide reference chains to underlying Links. The distinction between the quantity subclasses parameters and variables is that parameters always remain fixed at a given value throughout a simulation, while variables, naturally, vary.



Schema 2

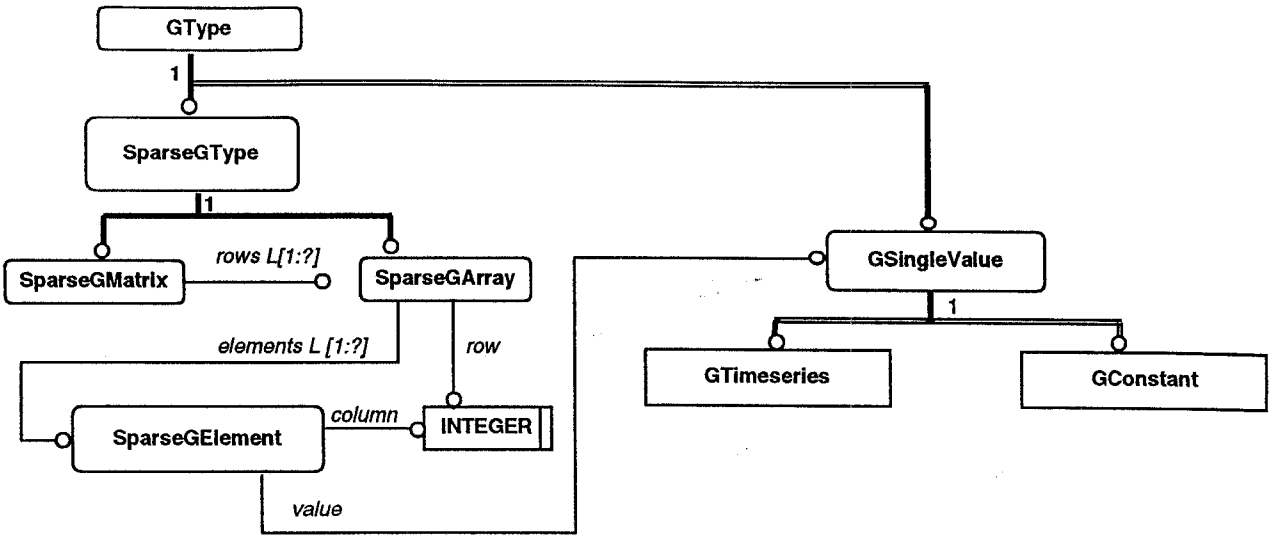
A quantity has a type (the NMF QUANTITY_TYPE referred to), a name, a dimension (if non-scalar), and a current value. Variables also have an initial value, which holds the state at the beginning of a simulation for dynamic (or state) variables and an initial value guess for algebraic variables. Discrete variables is a provision for future development of discrete time NMF models and is not currently used.



Schema 3

Continuous variables also have an optional flag given which, if present, indicates that a variable, or selected parts of a field variable, are to be kept at a given value throughout the simulation.

Schemata 3 and 4 specify storage structures and stored elements for current and initial values (schema 3) and for given flags (schema 4).



Schema 4

Values may be stored in either a sparse matrix storage structure or in full matrices (or ditto arrays). Initial values are generally stored in a sparse structure where only exceptions from the default value are listed.

Since the great majority of variables are calculated, the given flag is stored in a sparse structure as well. The flags themselves are either a reference to a time series of values (not specified in detail) or a GConstant symbol, indicating that the variable is to be kept at its initial value throughout the simulation.

6. CONCLUSIONS AND FUTURE WORK

Our present work suggests that EXPRESS is suitable for modelling of many of the data structures that are relevant for continuous simulation of modular systems. If not incorporated into the STEP effort, a continuous simulation language standardization project could certainly operate in a similar fashion and use many of the same methods and tools.

Next on the agenda will be to test the functionality of the suggested data structures by writing a parser for, initially, IDA system model descriptions.

References

Augenbroe, G, and F. Winkelmann, 1991, Integration of Simulation into the Building Design Process, proc. of Building Simulation '91 , Nice, France, International Building Performance Simulation Association

Augenbroe, G. (ed), 1993, COMBINE Final Report, CEC-DG XII-JOULE

ASHRAE 1993. Invitation to Submit a Research Proposal on an ASHRAE Research Project: 839-TRP Development of a Component Model Translator for the Neutral Model Format, American Society for Heating, Refrigerating and Air-Conditioning Engineers, Atlanta, GA

Buhl, W.F., E.F. Sowell, and J-M Nataf, 1989. Object-oriented Programming Equation-Based Submodels, and System Reduction in SPANK, proc. of Building Simulation '89 , Vancouver, BC, International Building Performance Simulation Association

ISO TC 184 1993. The STEP Standard, draft international standard DIS 10303, continuously since 1992 published in several different parts

Jeandel A. 1994, Personal communication

Kolsaker, K. 1994a. NEUTRAN-supported NMF Enhancements, presented to the TC 4.7 NMF Ad Hoc Subcommittee at the ASHRAE winter meeting 1994

Kolsaker, K. 1994b. Simpler NMF Description of Advanced Models Using Hierarchical Modelling and Data Abstraction, to be presented to the TC 4.7 NMF Ad Hoc Subcommittee at the ASHRAE annual meeting 1994

Lorenz F. 1987, Reflections about Representation Methods, proc. workshop on the future of building energy modelling, Ispra, Italy, Nov. 1987, CEC EUR 11603 EN PREPRINT, May 1988

Lorenz F. 1990. Brief Description of the MS1 (Modelling System 1) Project, private communication

Lorenz F., 1994, Comments on the Neutral model Format, presented to the TC 4.7 NMF Ad Hoc Subcommittee at the ASHRAE winter meeting 1994

Nataf J.-M. 1994, Translator from Neutral Model Format to SPARK, draft paper presented to the TC 4.7 NMF Ad Hoc Subcommittee at the ASHRAE winter meeting 1994

Pelletret, R. 1994, Personal communication

Pelletret, R., S. Soubra 1994b. Standardizing Model Documentation - The PROFORMA Experience, presented to the TC 4.7 NMF Ad Hoc Subcommittee at the ASHRAE winter meeting 1994

Sahlin, P, E.F. Sowell 1989, A Neutral Format for Building Simulation Models, proc. of Building Simulation '89 , Vancouver, BC, International Building Performance Simulation Association

Sahlin, P. 1991, IDA - a Modelling and Simulation Environment for Building Applications, Swedish Institute of Applied Mathematics, ITM Report no. 1991:2

Sahlin, P., A. Bring, and E. F. Sowell, 1994. The Neutral Format for Building Simulation, Version 3.01, Swedish Institute of Applied Mathematics, ITM Report no. 1994:2

Sowell, E.F. 1994. A Proposal for Hierarchical Submodels in NMF, to be presented to the TC 4.7 NMF Ad Hoc Subcommittee at the ASHRAE annual meeting 1994