TRANSFERRING SIMULATION TECHNIQUES TO END-USERS APPLICATION TO TRNSYS

Roger Pelletret, Souheil Soubra, Werner Keilholz
CSTB - BP 209
06 904 Sophia Antipolis Cedex
FRANCE
Phone : (+33) 93 95 67 00
Fax : (+33) 93 95 67 33
e-mail : iisibat@cstb.fr

ABSTRACT

The objective of this paper is to present an overview of the latest developments of the CSTB R&D program 'Intelligent Simulation Environment (ISE). This project tackles the problem of facilitating the transfer of technologies from the research centers to engineers in consulting offices dealing with building related issues (indoor air quality, energy consumption, acoustics, structural analysis, etc.).

The ISE can be seen as an open, extensible construction set that is well suited to team work and that allows one to encapsulate existing simulation codes into one homogeneous environment with minimum cost. This is done through the use of object orientation and innovative software techniques (layering of the set of classes, automatic class generation, etc.). The main lines of the ISE architecture will be described and examples of use for the integration of new simulators will be given.

The ISE focuses on the end user aspects. Three user groups are envisaged- model developers (users who invent new components), project developers (users who describe a simulation project by assembling components) and analysts (users who will make economic and technical studies of existing projects). These user groups will be detailed and the ISE features to manage the team work around simulation projects will be presented.

INTRODUCTION

In the field of building and HVAC design, many simulation tools are available. These tools are generally developed by researchers and are usually rather powerful. By using these tools throughout the design process, the building professionals (designers, constructors, property managers, etc.) can improve their productivity and increase the quality of the buildings and the systems. Nevertheless, too few building professionals use simulation tools. This is probably due to the following reasons:

- simulation tools are deemed to be sophisticated but difficult to use (e.g. input data is too lengthy, simulation projects have an enormous number of components and a huge number of connections, programs are not user friendly, etc.);
- there is no direct mapping between model parameters and the data that is most commonly used by professionals (catalogue data, geometrical descriptions, etc.);
- for a well-defined simulation goal, it can be difficult to find the appropriate model. Information about the model (underlying hypothesis, applicability range, etc.) is seldom available;
- not enough time and money are available during the design phase of a building project. Because of this, professionals state that they cannot afford to use simulation tools in this phase and therefore continue to use simplified calculation tools and pre-defined solutions. These professionals still need to be convinced that increasing the duration of the design phase and using powerful simulation tools should allow them to find innovative solutions that can make up for the extra time spent in the design phase and turn out to be profitable for the overall building project.

The CSTB R&D project entitled 'Intelligent Simulation Environments' tries to come up with solutions for the above mentioned issues in order to facilitate the transfer of technologies (i.e. simulation codes) from the building related research centers to the building professionals.

1. SOFTWARE ARCHITECTURE

The global architecture of the ISE is structured in three layers. The structuring in layers of the software components along with object orientation allows one to re-use a large number of existing components when integrating a new simulator and therefore decrease the development costs.

1.1. The Core layer

The core is the level of the ISE where generic classes commonly needed to construct a simulation environment can be found e.g. variables, units, dimensions, models, libraries, etc. Each of these entities has associated methods thus allowing instances to communicate by message passing, respecting the ISE Application Programmer's Interface (API). Most objects, for example, 'understand' a message called 'prompt' which opens an application allowing the user to interact with the object on the screen. For example, sending the 'prompt' message to a variable will yield a window where the user can consult the name of the variable, its associated magnitude and default unit, the min, max and default value, etc.

The core contains important classes that assure the extendibility of the system. For example, the class 'ise-simulator' is used to group together all particularities of the simulator currently used in the environment. Several instances of this class can exist at the same time therefore allowing several simulators to be used in the same environment.

1.2. The Simulator Specific layer

This layer is intended to allow developers to introduce new classes that represent the features of the simulator that needs to be integrated in the environment. To do so, the developer must subtype the classes of the core, thus inheriting their default behaviour. In case this default behaviour is convenient, no more development is needed. Otherwise, if customised behaviour is needed in order to adapt the environment to some specifities of the simulator, the developer redefines the existing methods and adds new methods if needed.

1.3. The User defined layer

This level contains classes that represent physical models. These classes are indirect subclasses of the class 'ise-model' that is defined in the core and subtypes of the class 'ise-<name-of-simulator>-model' that is described in the simulator specific layer. The classes in the user defined layer are not defined by developers but are automatically generated by the system, guided by the user interactions (e.g. a graphical editor where the user enters the names of the variables of the model, a pointer to a set of equations or to a subroutine, etc.).

For example, if we want to work with the simulator TRNSYS [TRNSYS 90], the developer who implements the environment must introduce a new class into level 2 and call it 'ise-trnsys-model'. After that, the author of the models will create a new model in the TRNSYS library called 'zone' using an editor that allows him to describe the Parameters / Inputs / Outputs / Derivatives of the model and to define a pointer to the Fortran subroutine representing the TRNSYS type. This information is then used by the system in order to automatically generate a ready to be used class 'zone' that will have all the features described by author of the model plus all the features inherited by the father class 'ise-trnsys-model' and the grandfather class 'ise-model'.

The class generation mechanism is essentially used by ISE developers who will be able to parameter all the model related features like for instance the instanciation method but is also used by the authors of the models since it allows use of inheritance of variables between the models. This is simply done by attaching new models to existing ones in the models library. This means that the son model will inherit all the features of the father model and can, in addition, have specific features.

Inheritance between models is very useful to better structure the models since it allows the author of the models to describe at a high level a behaviour that is common to several models (e.g. storage tank) and create submodels (e.g. storage tank with variable inlet, storage tank with fixed inlet, etc.) by simply describing their specifities since they already inherit the common behaviour.

2. USER INTERFACE OVERVIEW

Another important feature of the ISE is to facilitate access to the simulation tools used for Building Performance Evaluation (BPE). To achieve this goal, two means are used within the ISE :

- graphical and user friendly front-ends;
- the separation between different types of users.

It is indeed a major goal of the ISE to provide a sophisticated graphical environment to allow users to access and modify the data managed in a simulation environment in a straightforward and user-friendly manner. Nevertheless, user-friendly interfaces are becoming a must and BPE should not be an exception. But the ISE offers more on the user interface level than just another graphical front-end around existing BPEs. One of the major problems that limit the spreading of BPE in building related professions is the fact that the same environments are usually supposed to be sufficient to deal with distinct facets of the modelling/ simulation process. This means that users that do substantially different tasks such as create new models, data input, result interpretation, etc. are supposed to be the same or, a the least, have the same needs and problems.
The ISE distinguishes four different types of users and offers for each of them a front end that is adapted to their tasks and needs.

2.1. Application programmers

Application programmers will integrate a new simulator in the simulation environment using for this task the core level (cf. §1.1). The application programmers will reuse existing classes or introduce new classes. These developments will be done with an object oriented extension of LISP [ILOG 91]. The application programmer has access to text files generated automatically that describe the classes and methods of the core. These text files are also used to generate the reference manual of the core. In a future version, this documentation will be available on-line using hypertext browsers therefore allowing the user to navigate from one class or method to another.

2.2. Model developers

Model developers are users who invent and produce new components stored in model libraries. To do so, the ISE provides graphical editors and tools in order to describe and test the introduced models. For instance, the model developer can load a dictionary that contains the units and magnitudes most frequently used in the studied domain (e.g. thermal analysis, air flow modelling, etc.) and relate the variables of his model to magnitudes in the dictionary therefore allowing automatic translation from a unit to another equivalent unit.

This model related information will then be used by the system to automatically generate the associated class (cf. §2.3). The editors used to describe the models are also used to generate the model related documentation.

Model Documentation is of paramount importance in order to produce reliable software components (i.e. models) that can be used as industrial components. Today, two major standard methods for documenting basically continuous models exist around the world: the Proforma [Dubois 88] and the Neutral Model Format - NMF [Sahlin and Sowel 93].

The actual version of the ISE uses the Proforma while the next version of the ISE will use the future version of NMF[1] which will combine the advantages of the Proforma[2] and of the NMF[3]. This will allow us to integrate within the ISE automatic translators that enable model developers to easily upload and test models found on the network (for example, using World Wide Web hypertext browsers) in their own environment models.

2.3. Simulation Engineers

Simulation engineers are users who model real buildings, using the components that are stored in the model libraries (cf. Figure 1). No programming skills are required at this level. Simulation engineers should have modelling skills that allow

---

[1] Pending formal standardisation, ASHRAE (American Society for Heating, Refrigerating and Air-conditioning Engineers) has formed an ad-hoc subcommittee that maintains NMF.

[2] The main aim of the Proforma is to facilitate the transfer of knowledge related to the models. This makes the models re-usable in a correct way since the physics embedded in the models, as well as the underlying hypotheses and limitations of the models, are presented in an explicit and standard way.

[3] The main aim of the NMF is to provide a neutral format (i.e. models are expressed in a general manner independent of any existing or planned environment) to describe the models. NMF defines a formal syntax (using Backus-Naur Form) for stating the set of differential/algebraic equations related to a model in an unambiguous way. This makes automatic translation of the equations possible. Equations are translated into subroutines that can be directly used by various target environments.

them to choose the models in the library that are adapted for the simulation goal (to do so, they can consult the model related documentation in order to be aware of the validity domain and underlying hypotheses of the models they are selecting) and assemble these models to represent a real project (e.g. building, HVAC system, etc.). Assistance is offered to the user in the process of connecting the models: linking rules will check if the two connected variables share the same physical magnitude (e.g. air temperature, mass flow, etc.). Once the assembly is constructed and tested, it can be saved in a project library as a ready to run project.

The simulation engineer can lock some parts of the projects (entire models, variables, etc.). In that case, the locked parts will not visible to the end users who can only access the unlocked variables. This feature allows the simulation engineer to 'filter' the information related to his project and make sure that parametric studies carried out will not include critical or non-relevant variables.

Simulation engineers can also use the ISE Application Construction Kit (ACK) to create end-user specific applications of the utmost simplicity without requiring any programming at all.

The aim the ACK module is to allow simulation engineers, once they construct and test the assembly of models underlying a simulation project, to 'draw' a scheme representing the real system modelled (cf. Figure 2).

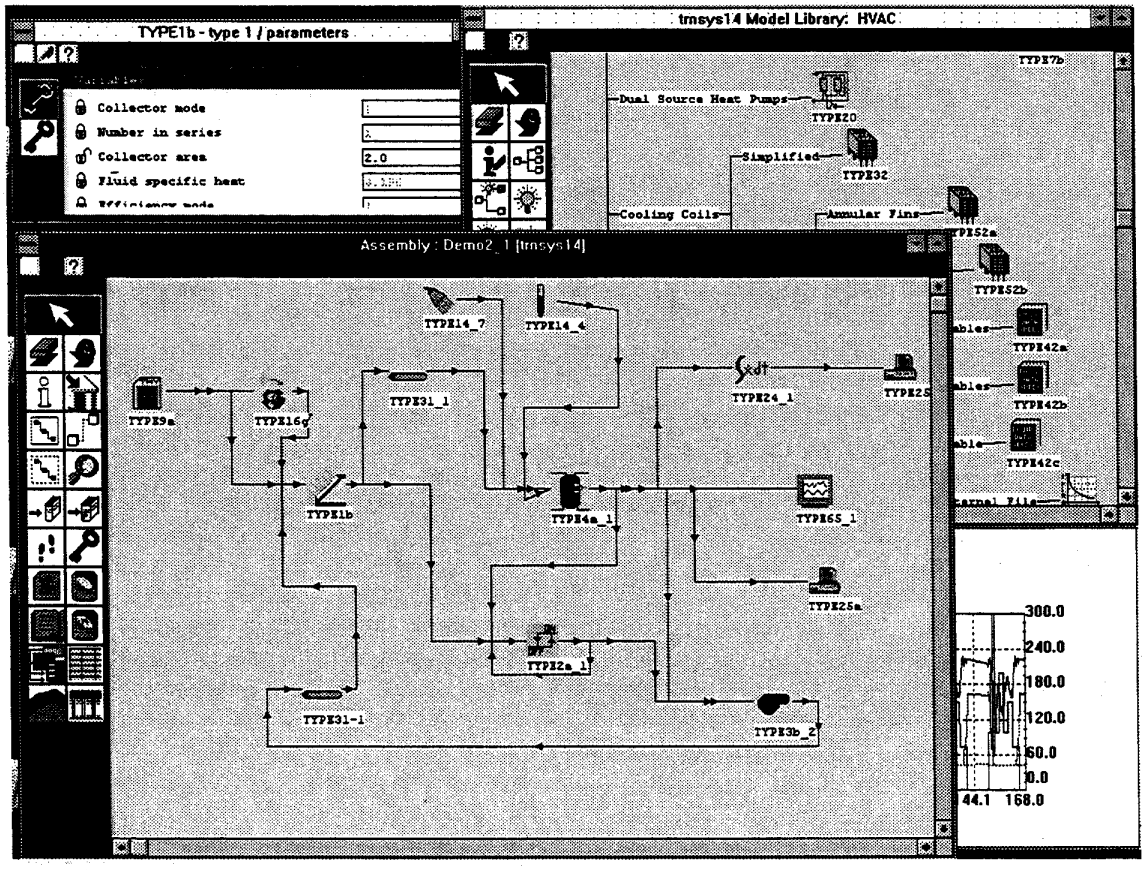

Figure 1: The ISE front-end as viewed by a Simulation Engineer using TRNSYS.

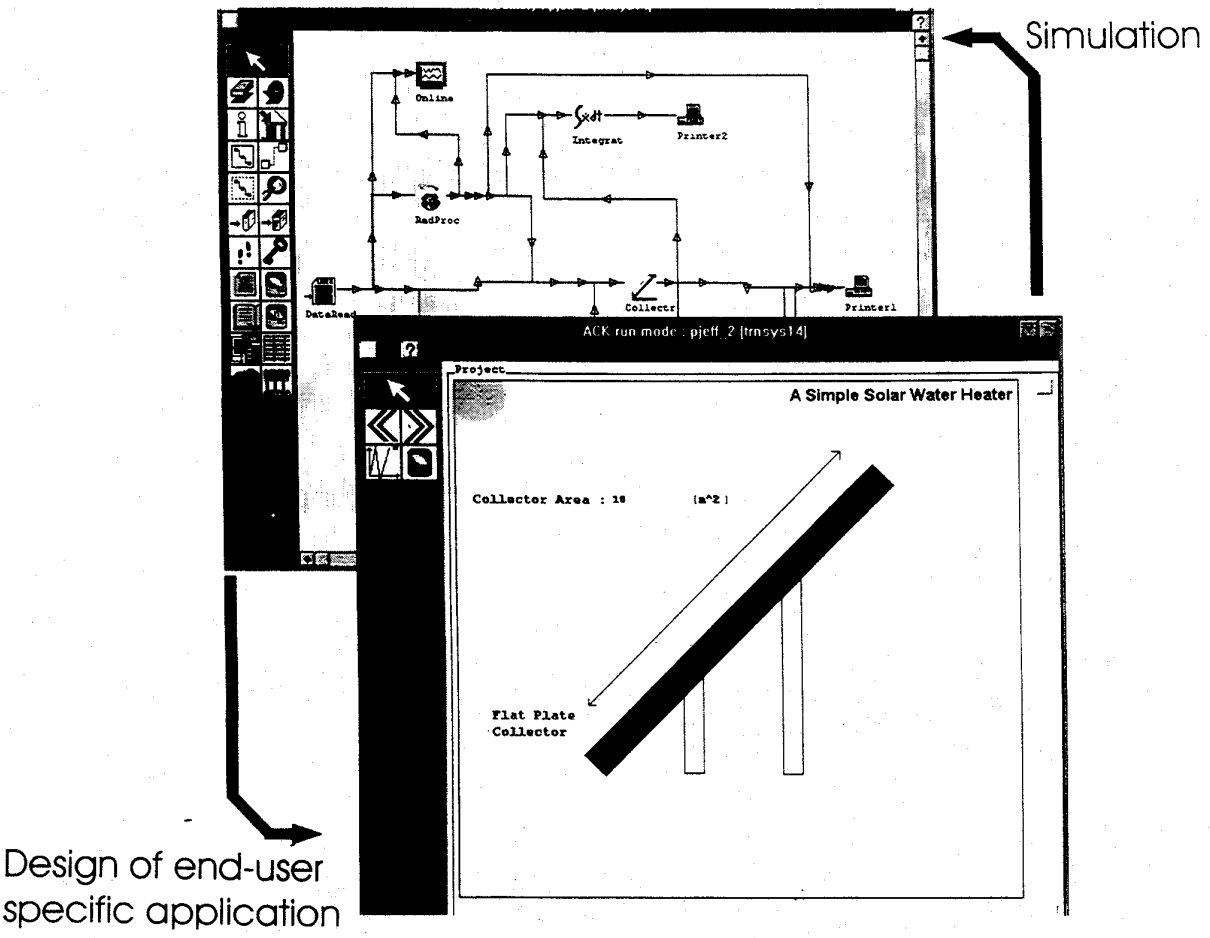

Figure 2: The Application Construction Kit (ACK).

Screens of the to-be application can be designed with ACK using a graphical editor. Variables can be placed in this drawing along with labels explaining their meaning. From this input, the system can generate a stand alone application that can run without the simulation environment. This allows simulation engineers to design simple and user-friendly applications that are meant to be disseminated among non-expert users even when the underlying simulation project is rather complex..

2.4. Analysts

Analysts are users who make economic and technical studies (i.e. parametric analysis) through stand alone applications developed with the ISE ACK. They do not need any programming nor modelling skills but only need to understand the physical phenomena in play in order to give values to the variables and interpret the results of the simulations. Analysts are guided when giving values to the parameters by checking functions which verify that the values entered are within the validity range of the variable as it has been described by the author of the model.

Finally, all four types of users have access to an interactive hypertext on-line help that describes the different procedures and tools (cf. Figure 3).

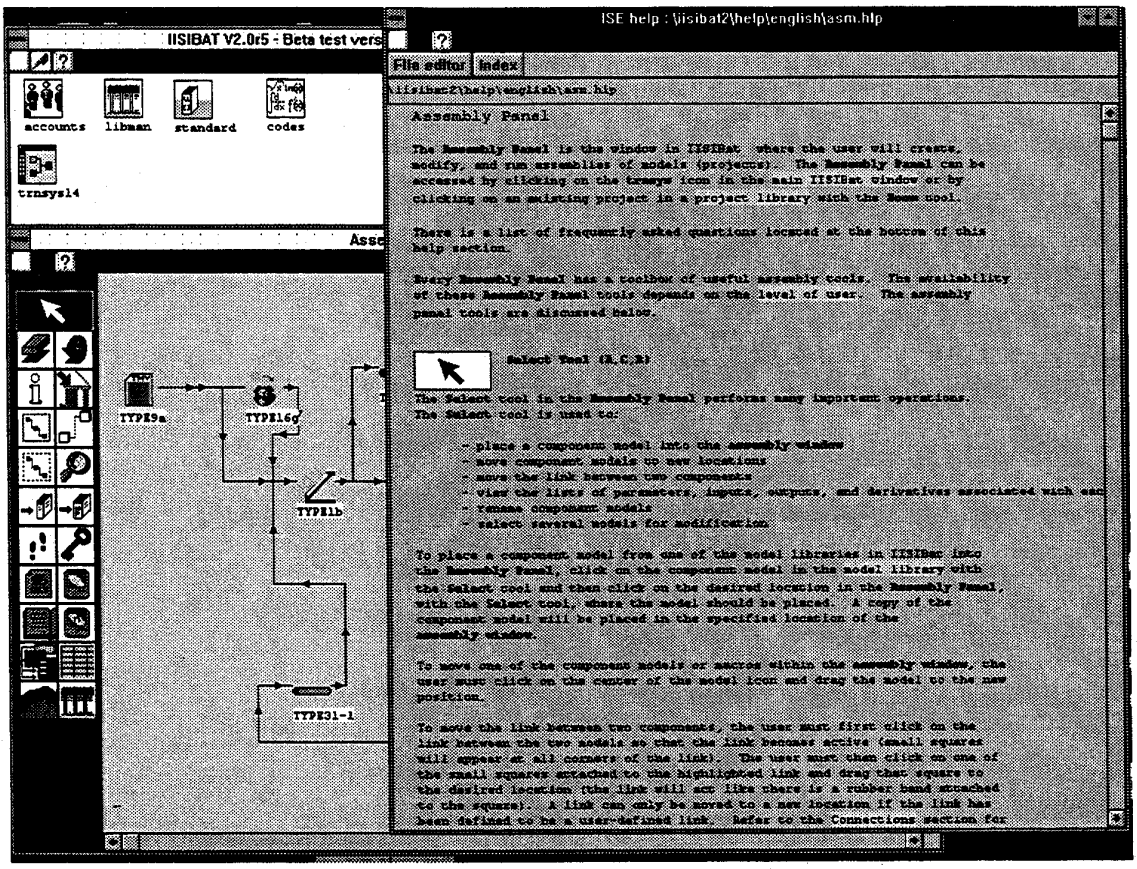

Figure 3: The hypertext on-line help system.

THE IMPLEMENTATION ISSUES

From the computing point of view, the development of the ISE is based on an object oriented approach: we use Le_Lisp which includes the object oriented Language MicroCeyx and the graphic interface generator Aïda [ILOG 91].

The ISE runs on machines with UNIX as Operating System and X11 as window manager; it also runs on PC/DOS under Windows.

CONCLUSION

The ISE project aims to help in transferring technologies (i.e. simulation codes) from building related research centers to consulting engineers.
The first application with the ISE was developed around TRNSYS[4] [TRNSYS 90].

This application is now available with the version 14 of TRNSYS.

Another application is being developed in the frame of Annex 23 of the IEA (International Energy Agency). This application is intended to encapsulate COMIS[5] [Feustel 89] and will be available in 1996.

COMIS has also been coupled to TRNSYS (i.e. a new TRNSYS type - TYPE 57 - representing the COMIS Infiltration Model is now available). Therefore, it is now possible to use the coupling of TRNSYS/COMIS in order to make simultaneous evaluation of building energy performances and related air flow patterns.

[4] TRNSYS is a thermal simulation code developed by the University of Wisconsin - Madison.

[5] COMIS (Conjuction of Multizone Infiltration Specialists) is a Multizone Air Flow and Pollutant Transport model developed at the Lawrence Berkeley Laboratory.

REFERENCES

[Dubois 88]
A.M. Dubois. 1988. A draft of a component model library manager, the "Modelotheque": analysis of an approach and first results. In Proceedings of the 3rd international symposium "Systems Analysis ans Simulation". Berlin. September. 1988.

[Feustel 89]
H.E. Feustel et al. The COMIS Infiltration Model. LBL. Berckley. 1989.

[ILOG 91]
ILOG: Le-Lisp Version 15.26. Reference Manual. Gentilly. Novembre 1991.

[Nataf and Winkelmann 92]
J-M. Nataf and F. Winkelmann. 1992. Automatic Code Generation in SPARK. : Applications of Computer Algebra and Compiler-Compilers. Research Report. Simulation Research Group. Lawrence Berkeley Laboratory.

[Pelletret and Soubra 93]
R. Pelletret, S. Soubra. 1993. The Concept of Intelligent Simulation Environment. In Proceedings of the IBPSA conference. Adelaide. Australia. August 1993.

[Pelletret and Soubra 94]
R. Pelletret, S. Soubra. 1994. Standardizing Model Documentation. The Proforma Experience. ASHRAE T.C. 4.7 Ad Hoc NMF Subcommitee. New Orleans 1994.

[Sahlin and Sowel 89]
P. Sahlin, E. Sowell. 1989. A Neutral Model Format for Systems in Building Energy Simulation. In Proceedings of Building Simulation '89. IBPSA. Vancouver. Canada. June 1989.

[Sahlin 93]
P. Sahlin 1993. A Neutral Model Format for Building Simulation Models. In Proceedings of the IBPSA conference. Adelaide. Australia. August 1993.

[Soubra and Pelletret 93]
S. Soubra, R. Pelletret. 1993. Intelligent Simulation Environments: A first Application to Building Construction. In Proceedings of SCS '93. Lyon. France. 1993.

[TRNSYS 90]
TRNSYS. A Transient Simulation Program. Solar Energy Laboratory. Unviversity of Madison-Wisconsin. Madison. Spetembre 1990.