An object-oriented architecture and simulation platform towards faster and portable SIMBAD simulations

Guillaume Ansanay-Alex

An object-oriented architecture and simulation platform towards faster and portable SIMBAD simulations

Guillaume Ansanay-Alex[1]*

[(1)] Centre Scientifique et Technique du Bâtiment, Marne-la-Vallée, France – *guillaume.ansanay@cstb.fr*

1. ABSTRACT

The new, 6.0 version of SIMBAD (SIMulator of Building And Devices), the energy efficiency oriented library for the construction of dynamic building models in the Matlab/Simulink environment, is currently developed at CSTB. This simulation tool provides the necessary components to simulate dynamically at a small time scale (less than a minute), or emulate in real-time, complete buildings along with occupants and systems, as well as to develop and evaluate building control strategies.

The two main evolutions in this new version of SIMBAD are the following: the rewriting of many essential models in an object-oriented fashion using the C++ programming language, and the creation of a C++ simulation platform including algebra, numerical analysis and thermodynamics tools. Every single modeling block can then be included in Matlab/Simulink through an S-function. It makes simulations much faster, and lighter in terms of memory load than with the Simulink-only model blocks from previous SIMBAD versions. This new version brings a centralized description of all component parameters into a single XML file, making easier the edition and translation (in particular from Building Information Models) of modeling data. The choice of the C++ programming language also makes possible the communication with external libraries for visualization and communication with acquisition cards.

In the future, the concomitant use of the simulation platform and model blocks will enable the linking of every SIMBAD component or group of components with visual BIM edition tools as an energy efficiency simulation plug-in, as well as the creation and diffusion of easy to use, dedicated standalone tools.

Keywords: building simulation, energy efficiency, modeling, HVAC

2. TODAY'S VERSION OF SIMBAD

The SIMBAD Building and HVAC Toolbox allows to develop and test HVAC control systems or control strategies.

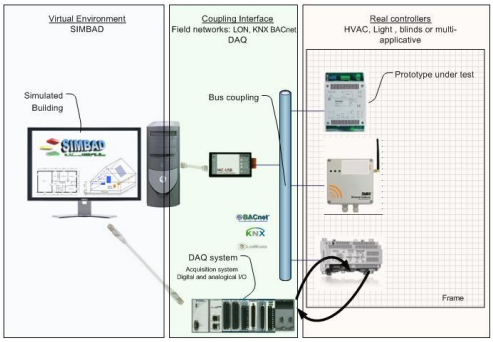

Figure 1: Test bench installation scheme for using SIMBAD as a virtual laboratory

It has been developed for years (Vaezi-Nejad, 1991) by the CSTB research team concerned with building automation and energy management as a set of Simulink model blocks. It is used in virtual laboratories for the test of real or simulated building control systems from the terminal level (thermal, lighting and blinds) up to Building Energy Management Systems. It includes a multi-zone building model which reads the building envelope description in a specific text file. Many HVAC systems are also included: heat production systems (heat pump, boiler), distribution elements (tubes, pumps, valves, manifolds, water storage tank, convergent and divergent), emitters (electric heaters, hot water heaters, fan coil units), lighting.

Figure 1 shows an example of setting up SIMBAD for coupling a virtual environment with real test benches: SIMBAD controls an acquisition card linked with the real product which is tested, and simulates a complete building as a virtual environment. Though it has been designed in the first place for a real-time use with real-life products, SIMBAD can also be used as a simulation tool to evaluate the energy efficiency of a fully equipped and controlled multi-zone building.

The currently distributed version of SIMBAD is numbered 5.0, and it is fully composed of blocks written using Simulink blocks and Matlab scripts.

3. IMPROVEMENTS OF THE NEW VERSION

3.1 Modelling toolbox updates

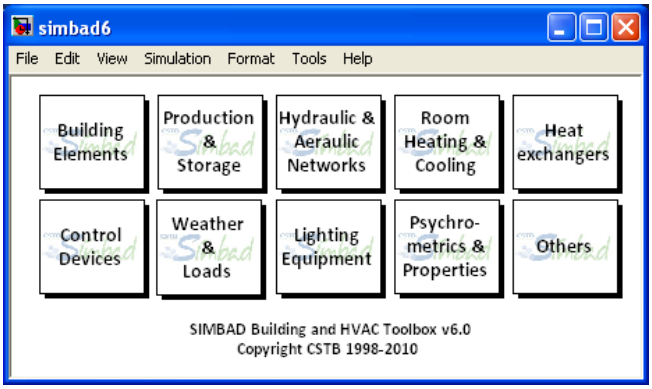

Figure 2: Main view of SIMBAD model blocks library as seen in Simulink

The main change in this new version is that a (still growing) great number of models have been doubled with their equivalents in the C++ language. But through this rewriting, we greatly improved many models and added some others. The new SIMBAD library then includes:

- an updated and optimized multi-zone building model, which now handles air quality evaluation with the CO_2 and humidity ratios, and reads its topology in an XML building description file (see section 4.1),
- models of renewable energy sources such as wind mills, photovoltaics and solar collectors,
- communication components for the use of SIMBAD in emulation : TCP protocol, data acquisition cards, OPC servers.

The use of the C++ language for the implementation of model blocks and of a hierarchical data structure for the building description enables a great interoperability with other dynamical simulation environments than the historically used Simulink (see section 3.2) and also enables the coupling with the wider and wider use of Building Information Models such as the Industry Foundation Classes (see section 3.3). It also makes SIMBAD simulations much faster and lighter in memory (see section 3.5).

Having developed SIMBAD modeling blocks in C++, we could also rewrite the simulation loop and time integration schemes necessary to build standalone tools: this enables us to plan the creation of energy efficiency oriented building simulation dashboards.

3.2 Interoperability: use SIMBAD models in other simulation environments

3.2.1 What is a dynamical simulation environment?

A dynamical simulation environment lets the user pick up modelling blocks in a library and connect them to build a complex system. It can then be simulated through a sequence of time steps. Some of the modelling blocks may have a dynamic behaviour, and then be described at each time step using current values of temporal variables. These are states we will call *continuous states*, and their time derivatives will be called *state derivatives*. When a block possesses such states, their initial values must be provided by the user.

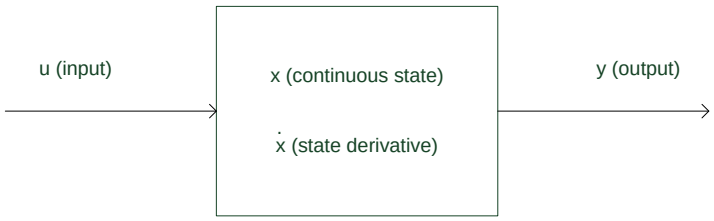

Figure 3: A dynamic model block as seen by the simulation environment

At every time step t_n :

$$y_n = f(u_n, x_n)$$

$$\dot{x_n} = g(u_n, x_n)$$

Then between times t_{n-1} and t_n :

$$x_n = \int_{t_{n-1}}^{t_n} \dot{x_{n-1}}\ dt$$

The simulation environment infers the execution sequence of modelling blocks from the whole system graph and runs each block after another in the simulation sequence, providing for each connection the output of a block as an input for another one. This kind of execution management is called causal simulation, and leads to series of independent computations. It is opposed to acausal simulation, which is able to consider groups of blocks as a whole, and leads to systems of equations solved simultaneously.

3.2.2 Examples of dynamical simulation environments

Simulink is a general tool for designing model-based dynamical systems, well known in many industrial applications, and has the ability to simulate models composed not only of discrete states but also continuous ones. Since 1994, a free software alternative, Scicos, has been developed by INRIA and gained a lot of maturity and users, as well in academic and private research teams. Scicos has now been rebuilt to become Xcos: it is distributed freely in the Scilab software, which is itself the free alternative to Matlab.

Let us at last evoke Trnsys, a tool widely used by engineering teams to simulate the transient performance of thermal energy systems. It is linked to the two previous dynamical simulation environments because it enables its user to create complex systems by linking model blocks, and then launch a simulation based on this model graph, though in its current version it does not provide integrated routines to cope with continuous states. It is specific to building simulation in the sense that it provides a rich library of building systems models usable in this environment.

3.2.3 Design generic blocks for every environment

These three dynamical simulation environments all provide a way to include user-defined models by calling external libraries. As more and more dynamical simulation environments are used in different research, development and studies fields, we believe that physical models are not any more to be built using specific, proprietary languages, if they are to be used at a large scale. That's why we first thought about what a generic model block should be like if it was to be used as an external model in any dynamical simulation environment.

With this aim in mind, we implemented two-layered block models:

The *Model Block* is not specific to any simulation environment: it is the core implementation of each physical model. It contains the functions which compute the outputs and state derivatives of a model, given its inputs and current states. It is invisible to the user.

A *Gateway* is not specific to any *Model Block*, but to the simulation environment: it's the source code file used to build the actual plug-in for each environment. At the time of this writing, we implemented prototypes of gateways for Simulink, Scicos and Trnsys.

For each dynamical simulation environment and model, a fourth layer, the *Block Mask*, can be defined (though in Simulink it is contained in the *Gateway* layer): it's the visual shell that can be dragged and dropped in the simulation environment. Its form is then thoroughly specific to each environment. Interface functions could though be used as a preliminary step to generate *Block Masks* of the *Gateways* from exported *Interfaces*.

3.2.4 Available gateways

As of today, gateways have been developed for the three dynamical simulation environments cited before, namely Simulink (see Figure 4), Xcos (see Figure 5) and Trnsys (see Figure 6). Further development is needed to give to this last one the same ease of deployment as the Simulink one, mainly because of a lack of specifications of the model mask templates (named *Proformas*). In the meantime, these model masks must be manually written.

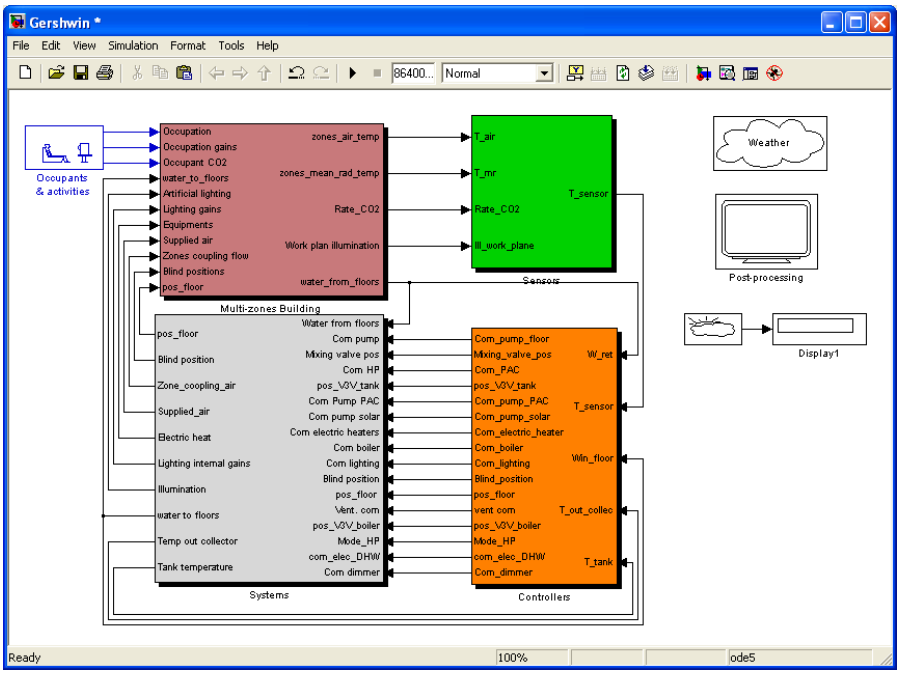

Figure 4: Example of a full building simulation system in Simulink using SIMBAD.

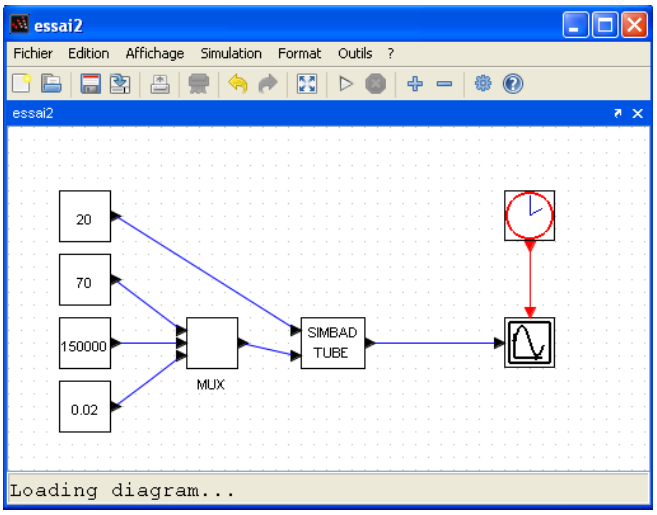

Figure 5: One of the new SIMBAD models used in the Xcos environment.

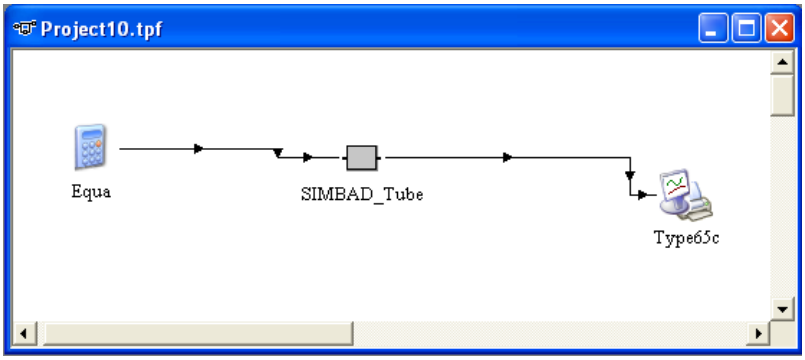

Figure 6: One of the new SIMBAD models used in the Trnsys environment.

3.3 Coupling with IFC Building Information Models

The Industry Foundation Classes (IFC) data model is a neutral and open specification of a Building Information Model developed by buildingSMART to facilitate interoperability in the building industry. Building Information Modeling is the process of generating and managing building data (building geometry, spatial relationships, geographic information, quantities and properties of building components) during its conception and life cycle.

We are currently working with the CSTB team ModEVE, which has a leading position in the use of IFC BIMs, and releases a C++ software development kit which enables to manage IFC models in a simulation tool. This team also develops a visualization and configuration platform named eveBIM. The aim we are pursuing is a more and more integrated way of designing buildings, running simulations and processing outputs, using the IFC as an operational data structure. At the time of this writing, the first step of this work: the exportation of an IFC building envelope description as a SIMBAD XML description is at a beta level.

3.4 Guaranteed validation and testing

SIMBAD has a long history in Simulink system modelling, and the existing models have been validated against physical measurements. Every model rewritten in the C++ language is then tested against the "old" model in Simulink to ensure a zero difference between model outputs, see Figure 7. It enables us to validate the non-regression of SIMBAD and, being driven by the tests, makes us improve the two versions, Simulink and C++, at the same time.

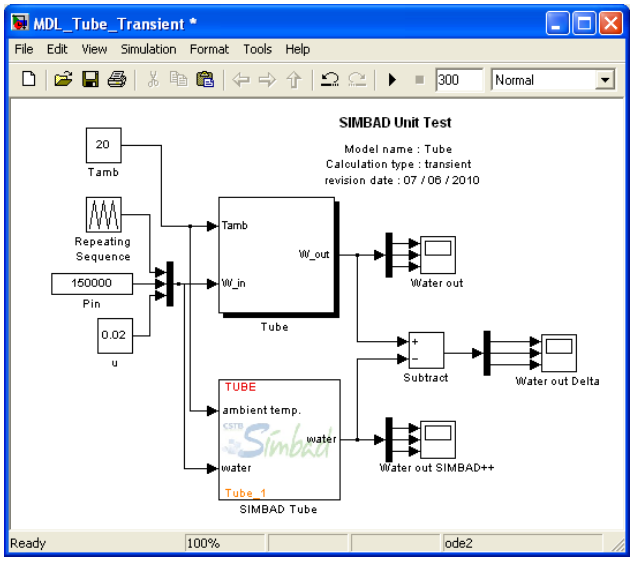

Figure 7: Unit testing procedure.

3.5 Lighter and faster simulations

The C++ components of SIMBAD have been thoroughly optimized in two directions. First, the simulations in previous versions of SIMBAD with Simulink-only blocks used huge amounts of memory, further growing with the number of time steps, due to the use of memory-consuming Simulink tools. The memory occupation of the C++ blocks now stays constant through the simulation and is very low: 34 Mb for an individual house with 11 thermal zones and the included HVAC systems.

We also spent some time optimizing the speed of SIMBAD simulations: on a laptop with a dual-core 2.8 GHz processor and 4 GB of RAM, using a time step of 1 minute and an order 3 time integration scheme, an annual multi-zone building simulation for the computation of the heating loads of a 11-zones house takes 10 minutes, and the annual simulation of this same house with all its equipments takes 33 minutes.

4. HOW THE USE OF SIMBAD WILL CHANGE

4.1 A new building description format

In the previous versions of SIMBAD, the information about the building was written in one text file, and for every other block, in fields of its Simulink mask. The aim that leaded us to design a generic architecture for model blocks usage naturally goes with the necessity to design a unified syntax for simulation and model parameters description. This means, adopt a normalized way to fill in the information about all the models, in only one place. We stated in the above description of block interfaces that simulation data structures for every SIMBAD block were built using the information read in a simulation description file.

The data model for the description of buildings and systems in SIMBAD is a hierarchical data structure using the XML syntax. It is currently divided in two sections. Under the *Building* node are described the building topology: storeys, rooms, walls, windows and masks, and the characteristics of building components. Wall types are for example defined by the set of their layers, each layer being described by its physical properties. Figure 8 shows how this information is organized.

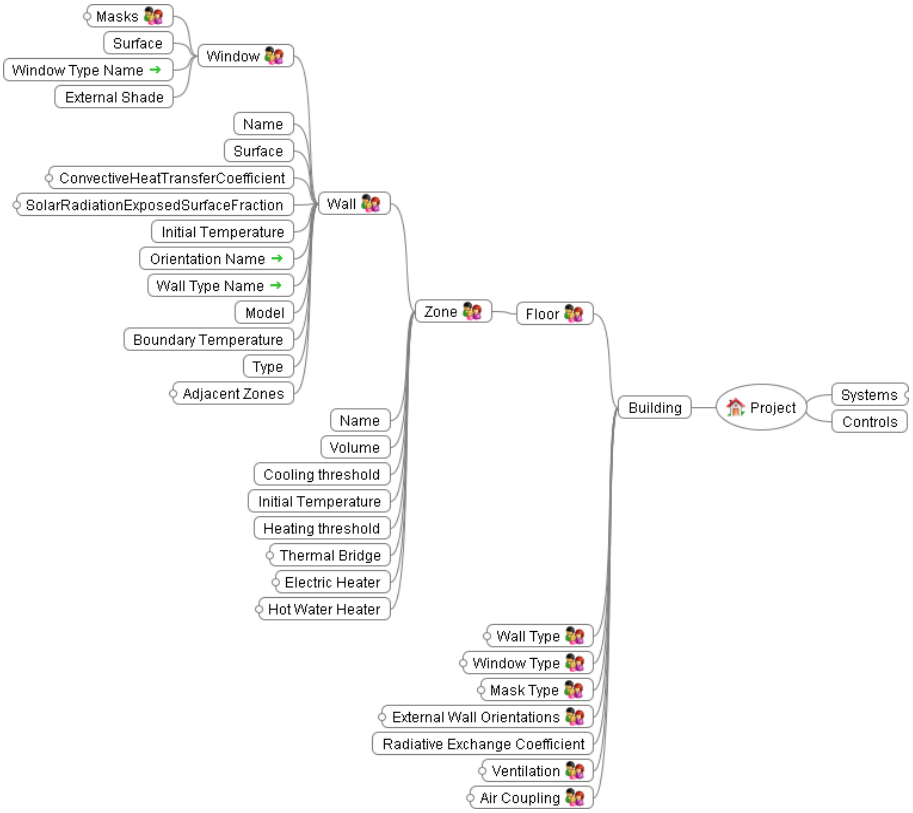

Figure 8: View of the "Building" section of the XML description

All HVAC systems that are to be included in the building are also described in the XML date structure, see Figure 9 for a partial tree structure illustrating the contents of this *Systems* part.

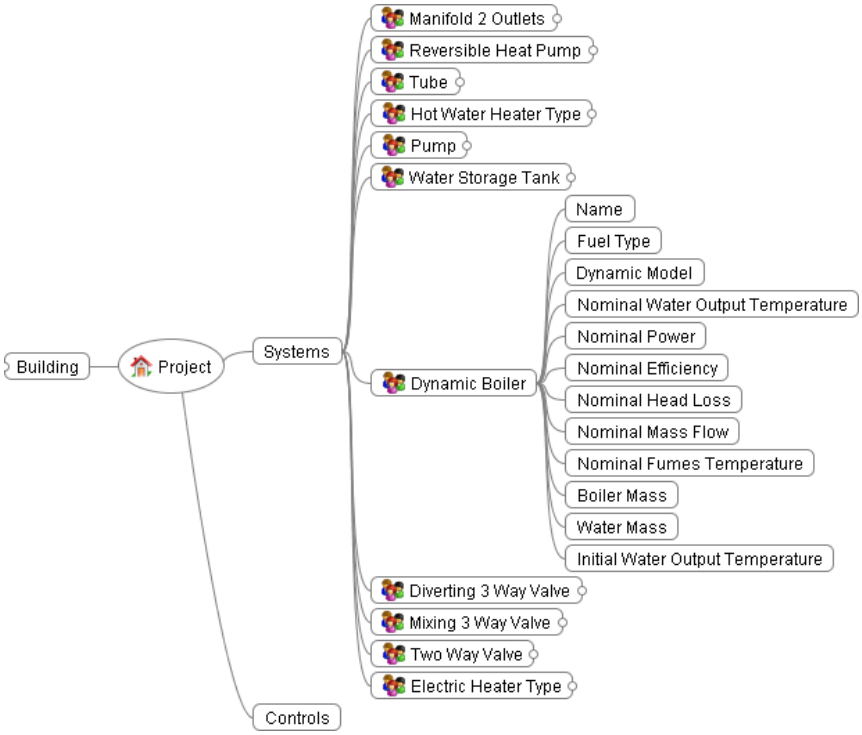

Figure 9: View of the "Systems" section of the XML description

Then, when a building or a system is present in the XML description file, it is very easy to include the corresponding model in the simulation environment. For example, in Simulink, see Figure 10, the generic gateway S-function makes use of the polymorphism permitted by the C++ language and only needs the path to the description and the object name to automatically morph the block inputs and outputs, along with their sizes and names in the block mask.

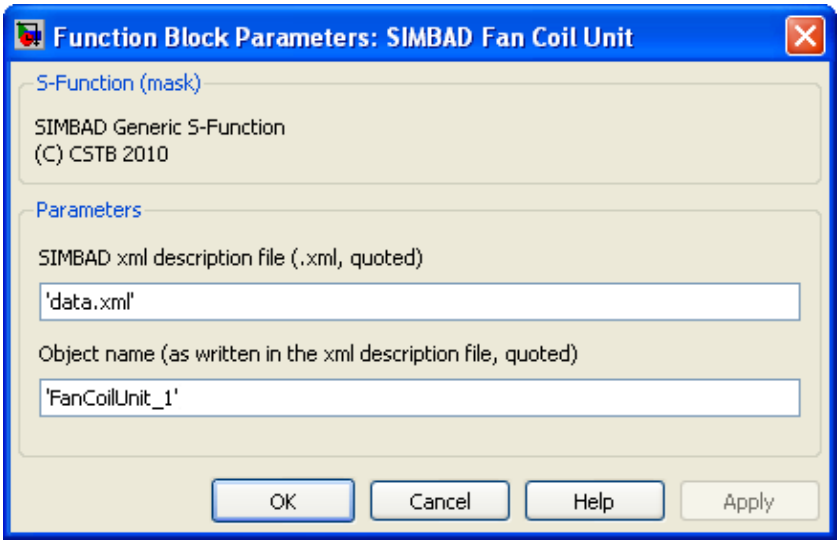

Figure 10: Simulink mask for a SIMBAD model block

4.2 A place for the SIMBAD users community

We will soon update the SIMBAD website, at the time of this writing still describing SIMBAD 4.0, to exchange a lot more with SIMBAD users and get their feedback on the use of SIMBAD. A SIMBAD user's mailing list is being activated to capitalize discussions, questions and answers regarding SIMBAD use and awaited improvements.

The new version of SIMBAD described in this paper, numbered 6.0, is planned for release at the end of 2011. If you feel interested, please contact the author at the SIMBAD e-mail address: *simbad@cstb.fr*.

5. CONCLUSIONS

We presented in this paper the new structure of the computation core behind the SIMBAD library. It brings a simplified and centralized way to describe a building and its components. It also enables the use of SIMBAD blocks outside of the Simulink dynamical simulation environment, and the creation of standalone tools, such as dedicated simulation tools and even dashboards. As a side effect, simulations have become much faster and less memory-consuming.

The next iterations of our work include using the possibilities offered by having a C++ library: we plan to implement graph-oriented methods to automatically configure systems and controls, and to standalone dedicated tools such as energy efficiency dashboards.

REFERENCES

Vaezi-Nejad H., Hutter E., Haves P., Dexter A.L., Kelly G., Nusgens P., Wang S, 1991, *The use of building emulators to evaluate the performance of Building Energy Management Systems*. Proceedings of the 3rd International IBPSA Conference, Sophia-Antipolis, France, pp. 209 - 213.

The Industry Foundation Classes specification, buildingSMART http://iai-tech.org/products/ifc_specification

Software Development Kit for reading and writing a Building Information Model defined in Industry Foundation Classes (IFC) format. http://www.osor.eu/projects/ifc-sdk

Simulink platform for multidomain simulation and Model-Based Design of dynamic systems http://www.mathworks.com/products/simulink/

Xcos, hybrid dynamic systems modeler and simulator http://www.scilab.org/products/xcos

Trnsys, Fortran program to simulate the transient performance of thermal energy systems http://sel.me.wisc.edu/trnsys/features/