AN OPEN ENDED MODULAR INTERFACE AND CONTROLLER LIBRARY FOR CLIM 2000

Kevin M. Murphy, and Francis Déqué
Électricité de France / Direction des Études et Recherches
Département Applications de l'Électricité dans les Bâtiments
Centre des Renardières, Route de Sens-Ecuelles, F 77250 Moret sur Loing France
Phone : 33 1 60 73 69 71 Fax : 33 1 60 73 75 79 Email : Francis.Deque@der.edfgdf.fr

ABSTRACT

The CLIM 2000 software environment [1] was developed by the Electricity Applications in Buildings Branch of the French utility company, Electricité de France. This software, which has been in operation since June 1989, allows the behavior of a whole building to be simulated. During the last phase of development, special attention, and hence research resources, were devoted to developing an open-ended software package, where new models of regulator could be added in a user friendly way. The new components that underwent development were: 1) directed at today's demands for analysis of the occupant's comfort; 2) the comparison of different heating or ventilation systems; and 3) the pertinence of a complex control system to meet these demands.

This paper addresses the latter of the three areas and the new strategy for control in building simulation is also presented. MATLAB connection allows the development of a new controller prototype. Once the prototype is available, it is transformed with *the same interface* into a controller model. This is achieved by the use of an open-ended FORTRAN/C computer controller simulation interface. These new modules or tools (denoted as *CLIM 2000 - LIBCREG.A connection)* provide the user with efficient analysis of system properties i.e performance and robustness. In addition, these tools allow the user to evaluate the complexity of the design versus its system performance, by comparing alternative designs in a user friendly environment.

Keywords : modeling, simulation, controller interface.

INTRODUCTION

Although a building is intuitively a simple structure composing of four walls, a floor, a ceiling, a door, etc., the model is anything but simple. Typically a building's model is composed of high order equations, multiple inputs (due to the many external influences), multiple outputs and in addition necessitates a structured computer interface in order to be constructed.

In recent years, our department launched a major project to develop an in-house multifaceted building simulator called CLIM 2000 [1]. Today, the simulator is used to perform economy studies on energy usage in residential housing. Along with the elementary component models representing walls, ceilings, floor, doors, etc., a data base of external weather data is integrated into the software. Hence, the user now has a powerful tool to efficiently test new concepts and develop more accurate and detailed models.

On the control side, the problem has been the lack of a representative model of the process and its interaction with the environment. Hence, given the combination of a representative model, and the wealth of today's controller design tools, practicable cost effective solutions are now within reach.

In order to close the gap between the availability of the model and the control design tools, a new project was launched in 1996 to connect CLIM 2000 to a control tool. This connection comprises of two stages. Firstly, MATLAB connection allows the development of a new controller prototype. Then, to simulate faster and to be independent of MATLAB, an external FORTRAN/C connection is developed. This includes the CLIM 2000 - MATLAB connection presented in [2,3].

The main objective of the study in [4] was to determine the feasibility of connecting CLIM 2000 via a FORTRAN/C interface to external applications and math libraries in order to develop comfort controllers for buildings. The constraint is clear: the new connection must provide a computer framework where the control design is focused on comfort control and less on programming of the basic controller design tools.

1 : BACKGROUND

1.1: CLIM 2000

CLIM 2000 [1] allows the behavior of a whole building to be simulated . The simulation is divided into three stages. Firstly the building is described by means of a graphics editor providing multi-windowed dialogue in the form of a set of icons. This is called Formal Type (TF), and represents the models chosen by the user. These elementary models, about one hundred in CLIM 2000, describe physical laws (conduction, convection, etc.) with a set of continuous equations. Secondly, these sets of equations are transformed into an electrical circuit representation and solved by ESACAP [5]. In order to simulate these equations, ESACAP uses a Gear's method with a variable sample time. Finally , a post-treatment of the simulation data is possible via a graphic tool.

1.2: CLIM 2000 - MATLAB CONNECTION

With the control problem, the laws are represented by a set of discrete equations with a constant sample time. In order to integrate control representation into building models, a connection between CLIM 2000 and MATLAB has been made.

The CLIM 2000-MATLAB connection is summarized in Fig. 1. The figure illustrates in block diagram form the essential elements of the existing connection.

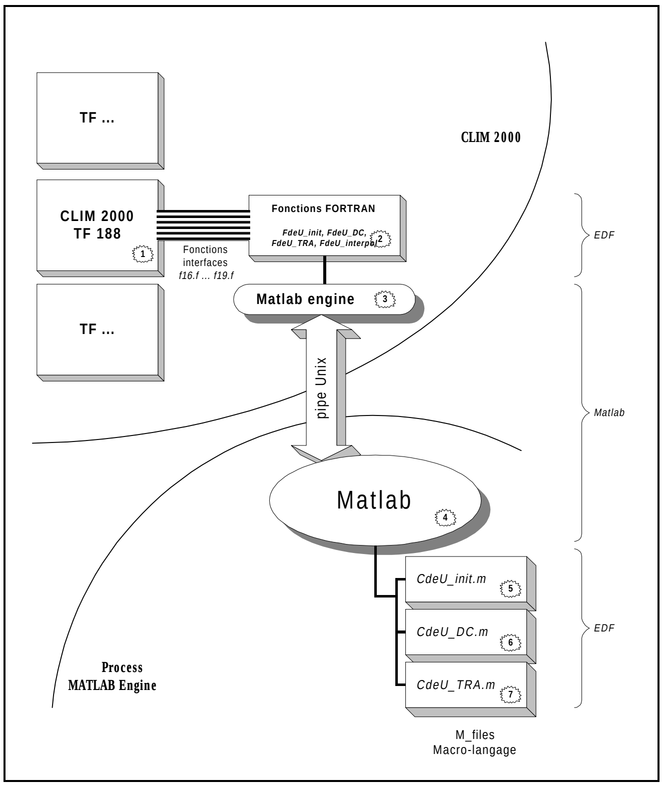


Figure 1: - Architecture CLIM 2000 - MATLAB

In brief, the CLIM 2000 application is a graphics-based interface where the user constructs the system using CLIM 2000's building component library, [1]. Then the user specifies the duration of the simulation, the external weather data etc.. and launches a simulation.

In order to use the CLIM 2000 - MATLAB connection, the user selects and implements the TF 188 (as described in 1.3) in the study in a standard way. As is shown in figure 1, the information (parameters) from this TF, as well as the computed control signal, are transmitted from CLIM 2000 to MATLAB by way of FORTRAN routines. These subroutines in turn engage system routines which launch the MATLAB application [6]. Then the UNIX pipe is used to transmit the information to memory where parameters from each application can be transferred and retrieved. This process is illustrated in the diagram by a transfer from the CLIM 2000 process to the MATLAB process. Finally, in the MATLAB process, high level Macro scripts, called M-files, are used which compute the control.

1.3: CONTROLLER USAGE IN CLIM 2000 - TF 188

The use of the Controller connection in the CLIM 2000 environment is essentially transparent to the user. This is achieved by the use of a new Formal Type (denoted as TF 188) and are coded and used as any other CLIM 2000 Formal Type.

At the user level via CLIM 2000's graphical interface, the input-output description of the TF 188 icon can be illustrated as in Fig. 2.

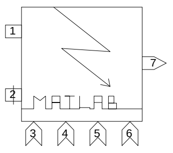


Figure 2: ICON representation of the TF 188 model.

The TF 188 is comprised of 6 input connections (or slots) and 1 output connection. In particular, slots number 1 and 2 are respectively the measured and setpoint variables. Slot number 7 is the output (typically a heating command). In addition, 4 extra slots are available to the user. Their exact definition depends on the specific control algorithm used.

As with other Formal Types, the specific use of the TF 188 requires the definition of internal parameters. In the present case, these internal parameters are:

Table 1 TF 188 parameters.

Symbol	**description**	**Range**
Nalgo	Algorithm number	>0
Tech	Sampling period	≥0
Umin	Minimum value of	]-inf,
Umax	Maximum value of	]-inf,
P1 to P10	Addition parameters	]-inf,

In summary, the 4 parameters (Nalgo, Tech, Umin, Umax) are fundamental to the operation of the TF 188. Their uses are as follows

Nalgo	-	selects a specific control algorithm,
Tech	-	specifies the sampling period,
Umin	-	sets the lower limit of the output,
Umax	-	sets the upper limit of the output, and

The remaining 10 parameters provide additional flexibility to the user. In particular, these parameters provide the user with additional algorithm options.

As with other Formal Types, all input-output signals and internal parameters can be saved automatically. They are specified in the standard way via CLIM 2000's interface, [1].

2 : CLIM 2000 - LIBCREG.A connection

CLIM 2000 - MATLAB connection allows the simulation of the building model using a library written with high level scripts in the MATLAB process. This connection is a powerful tool in a production of new models for control. However, this solution has its disadvantages. The simulations run slowly and royalties must be paid to the company that sells MATLAB. This is why Electricité de France has been developing the LIBCREG.A library. This library allows the connection of continuous equations of existing models to discrete laws of control models. These models are either generated automatically by MATLAB with the C-Generator tool or written in C or Fortran code by the developer. The user interface is the same as MATLAB connection.

2.1: LIBCREG.A ARCHITECTURE

Figure 3, represents an extended version of the CLIM 2000 - MATLAB connection presented above in Fig. 1. Specifically, in the present figure, a new link to external applications via FORTRAN/C is now provided. As is shown, the new capabilities now include the possibility of using the following additional MATLAB products:

1) MATLAB C - Compiler, and
2) MATLAB C - Math Library.

These products are used to develop C based binaries from MATLAB script files. For example, the use of a PI.m, [2] available in the CLIM 2000-MATLAB connection requires the main MATLAB application to run. However, using these new products, the original PI.m can be regenerated automatically into C using the C - Compiler and linked to the C - Math Library and run without the MATLAB application.

In addition, it is possible to develop C or FORTRAN based routines and link them to a new set of FORTRAN/C library routines.

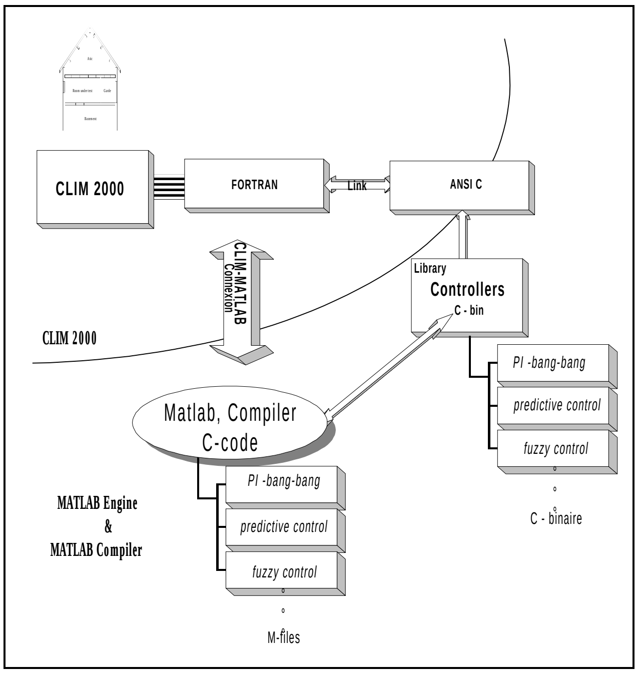


Figure 3: - Architecture CLIM 2000 - LIBCREG.A

2.2: <u>TECHNICAL LAYOUT OF THE LIBCREG.A FORTRAN/C INTERFACE LIBRARY</u>

In this section, the technical layout of the *LIBCREG.A* controller simulation interface library is presented. The main feature of the new connection is two-fold:

1) The developer can easily choose which, if any, external software products to use in developing his/her ideas.

2) The introduction of new controllers into the library requires minimal effort (less than ten lines of code).

The library is built on a foundation of object oriented programming concepts and therefore, future changes are easy to integrate into it. In this spirit, LIBCREG.A is constructed from four sub libraries which can be classified as either interface or computation by nature. In particular, the interface is the same as that used for MATLAB connection (Fig. 2 and Table 1). For the computation routines, it is important to further classify these into two sub categories:

Category 1) built-in routines, and
Category 2) external user defined routines.

In the former case, the developer can use these routines to perform essential computation tasks such as arithmetic matrix and vector computations. In the latter case, the developer can take advantage of other sources such as MATLAB products:

MATLAB C or C++ Compiler and,
the corresponding Math Library,

or established numerical software for solving linear and non-linear problems using:

LinPack, EisPack, , etc..., or Numerical Recipes.

Most of the above are now available in both C, or FORTRAN.

The five main sub libraries are:

Table 2 Libcreg.a.

	Libname	**Needs to be modified**
1)	*fdeu,*	*no*
2)	*interface,*	*no*
3	*cdeu,*	*yes (add 2 lines)*
4)	*control, and*	*yes (developer dependent)*
5)	*matrix.*	*yes (developer dependent)*

Essentially, *fdeu.a* and interface.a provide a means to retrieve and send data to the TF 188. Cdeu.a allows the choise of regulators. For example, the addition of two lines to three files in cdeu.a corresponds to providing a path to the new controller. The added lines are elseif, etc. The last two libraries (control and matrix) are developer-dependent in as much as the computations are performed in these libraries. More specifically, if the developer constructs a new controller, he places the main code in *control.a* and should put any additional subroutine or function dependencies into *matrix.a*. Hence, these two latter libraries are entirely at the control of the developer. In brief, the *control.a* library is typically a short file, in either FORTRAN or C describing the computation. That is, only basic computation should be performed here, with more complicated computations been performed in the matrix library. Therefore a minimal amount of effort is required to introduce an new controller into the library.

However, if the developer needs more computation power or for any other reason prefers to make the computations using other products, he/she needs to link these packages to the matrix library.

Next, a technical description of the *control.a* library is presented. This library contains all current FORTRAN and C controller computation routines.

Table 3 Current control routines.

NAlgo	**Control law**
1	PID Numeric
2	PI with bang-bang
3	Hystersis
4	RS - predictive control
5	GPC predictive control

For example, setting NAlgo = 2, causes the FORTRAN PI.f to be used in computing the control. Setting NAlgo = 4, causes the FORTRAN RS predictive controller (described below) to be used in computing the control. The final library, matrix.f contains low level computing routines for the *control.a* library. For example, intensive polynomial and matrix arithmetic operations are required to compute the predictive controllers R and S, and they are located in this library. As stated before, the complexity or number of function calls is developer -dependent.

The final library, LIBCREG.A contains four sub libraries:

libcreg.a = interface.a + cdeu.a + control.a + matrix.a

These four new libraries provide the user with a larger computing environment. For example, to include the C - Math library, the user needs to do the following:

1) add the new controller routine to *control.a*
2) add the new NAlgo to cdeu.a (c.g. ... elseif (NAlgo = ..) call new_contr(....))
3) regenerate CLIM 2000, including the three C - Math Libraries: libcreg.a libmcc.a libmatlb.a libtbx.a)

In a similar manner, a new controller requiring the MATLAB engine can be added. In short, new controllers, irrespective of computational environment, are added by following steps 1 - 3, where in the third step, the appropriate libraries (computational sources) are linked into the main program.

In this way, the FORTRAN/C connection is considered to be an open-ended controller simulation interface. The developer has a complete degree of freedom in designing new controllers and their subsequent library needs. Hence, we can see the possibility of using many products to prototype controllers quickly and efficiently. The final controller can then be either implemented using these products or re-coded using LIBCREG.A'S built in linear-algebra library, *matrix.a* or addition routines added to it.

In Fig. 4, the key options available to the developer are illustrated. Starting at the top of the figure, the developer can choose to use MATLAB connection to prototype a new controller. In this case, CLIM 2000 - MATLAB Connection can be generated by using the engine.lib library. As is also shown, the developer may choose to use the pure C or FORTRAN interface between the TF 188 and the rest of the library. In either case, the developer can further specify whether to use external products such as the MATLAB C Compiler and C Math Library, etc. All these possibilities are illustrated with the switches shown on the diagram.

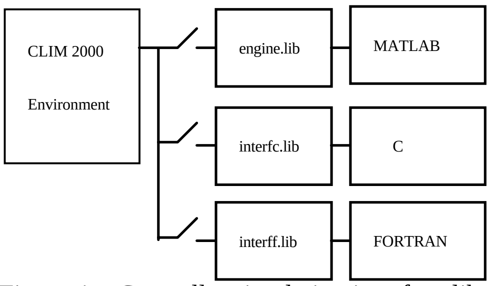


Figure 4: - Controller simulation interface library options.

3 : APPLICATIONS IN CLIM 2000

Figure 5 illustrates a single family residential house (Mi2). The figure represents the building components using circuit analogies. That is, each block corresponds to networks of resistors and capacitors, etc. and the connections points as nodes. As is seen, the room temperature is connected to the terminal (or node) 1. The setpoint is provided by the TF60 which reads a binary file containing the setpoint sequence. This in turn is connected to terminal 2. The output terminal is connected to the electric heater (TF 46). The remaining 4 input terminals are not used and are set at zero via a TF13 (constant output). The configuration is typically of TF 188 usage.

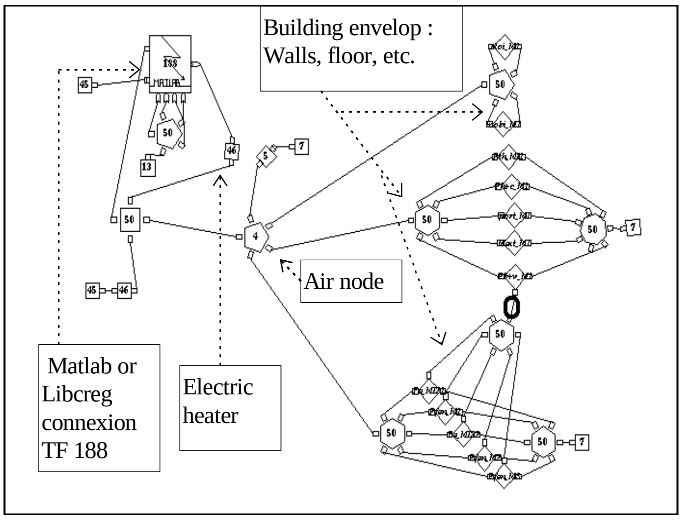


Figure 5: - CLIM 2000 graph of Mi2 with RS predict.

3.1 : APPLICATION OF MATLAB CONNECTION

Control objective :

The control objective is to obtain and hold a square-wave-like setpoint temperature in the Mi2 house. In particular, the setpoint is cyclic. It functions at a temperature of 14 °C from 12:00 PM until 8:00 AM, where the setpoint temperature is then 19 °C. Again at 6:00 PM, the setpoint is set at 14 °C until the following day where the cycle repeats, Fig. 6. Dashdot curves represent fuzzy controller response. Solid lines show the PI with bang-bang regulator.

As is clearly illustrated in Fig. 6, the regulation objectives are obtained with PI and Fuzzy control. However, it took approximately 2 hr. to arrive at the setpoint.

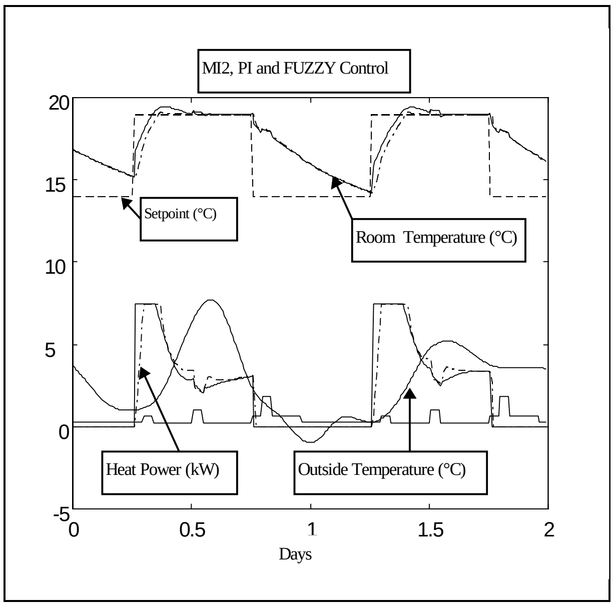


Figure 6 : Temperature and convector power evolution's

As a result, the occupants would probably consider the room to be too cold or uncomfortable. The comfort band is known to lie between one half degree of 19 °C. It is clear from the heating command, which correctly starts at full power that it is impossible to do better than this without additional power. That is, the temperature trajectory's rise time illustrated in the figure is increasing with a time constant of approximately 6 hours which is defined by Mi2's dynamics. Clearly, the solution is to either increase the heater power or launch the heater earlier.

3.2 APPLICATION OF LIBCREG.A IN CLIM 2000

Control objective:

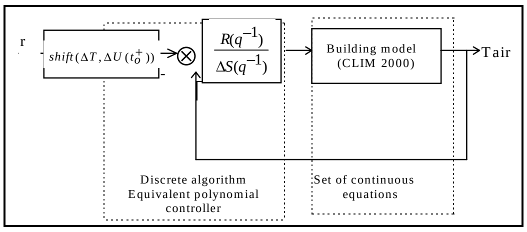
As was the case for the PI and fuzzy controller examples in the CLIM 2000 - MATLAB connection, the same TF 188 is used here to regulate the room temperature. The goal is to anticipate the setpoint change in temperature with a simple algorithm that can be implemented in a real controller. The model consists of simple recursive equations. In order to avoid royalties, the controller has been developed with the LIBCREG.A library.

Structure of the RS-predict controller:

The structure of the RS-predict controller is illustrated below in Fig. 7 where the parameters:

$R, S,$ and Δ - RS controller

are polynomial operators representing the RS controller. The details on computing these parameters are presented in [7] and [4] and are not repeated here.



r
$shift(\Delta T, \Delta U(t_o^+))$
$\frac{R(q^{-1})}{\Delta S(q^{-1})}$
Building model (CLIM 2000)
Tair
Discrete algorithm Equivalent polynomial controller
Set of continuous equations

Figure 7: Diagram of the RS-predict controller.

In brief, Fig. 7 represents in block diagram form the recursive equations used in computing the current control. In particular, the R, and S parameters are computed to optimize the step-response of the closed-loop system. The setpoint generator block:

$$shift(\Delta T, \Delta U(t_o^+))$$

where

t_o - time, (seconds)

$\Delta U(t_o^-)$ - reserve power of the heater,

ΔT - setpoint change in temperature.

is a FORTRAN/C implementation of an anticipative heater launch algorithm developed in [4]. Specifically, $shift(\Delta T, \Delta U(t_o^+))$ calls the following C functions from *control.a*:

Function Name	**Computes**
det_set.c, and	detects and stores future setpoint changes.
est-det.c.	returns an estimate of the time needed to drive the current room temperature to the future setpoint detected by det_set.c.

The value returned by $shift(\Delta T, \Delta U(t_o^+))$ is the value of the future setpoint such that the time to this future setpoint is equal to launch time estimate from est-det.c. The workability of this process stems from the high-gain nature of the RS- regulator developed in [7]. Hence, the room temperature is driven at the maximum rate of the open-loop plant.

In addition, it is important to point out that the RS-regulator is applied continuously throughout the process and hence, no special initialization is needed, as is the case when PI with bang-bang is used. Also, the stability nature (from simulation) of our new algorithm is substantially better in comparison to the PI with bang-bang case.

For the present example, a simplified model was used to compute the RS predict controller parameters and also used by the heater launching routines, the model of which is presented in [4].

The results of the simulation are summarized below in Fig. 8. The three curves are: the desired room temperature setpoint; the actual room temperature; and the heating command. As is clearly illustrated in the figure, the control objectives are obtained with very accurate regulation. Except for the initial day, the setpoint demands are achieved to well within specification at 8:00 AM. For the first day, the room temperature took a little more than 1 hour longer to arrive at the comfort setpoint.

This effect is due to the fact that when system is in equilibrium, as it is after a three day week-end, additional energy is needed to reach the setpoint. The additional energy is required to replace the energy dissipated by the walls during the week-end.

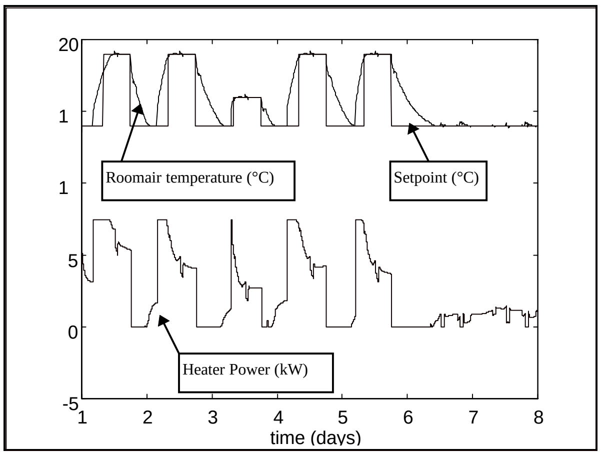


Figure 8: - Temperature and convector power evolution's.

Notice, in particular, that the heater is launched correctly at full power at t_{launch} and that the heater is applied at full power until just before 8:00 AM where it begins to decrease. The system perturbations for each of the days shown are due to the external air temperature, solar flux, etc... These factors acting on the building were continuously changing in level and the control algorithm automatically adapted the power level accordingly.

4 : CONCLUSION AND FUTURE WORK

In conclusion, this paper has illustrated the key aspects of the controller simulation with CLIM 2000. MATLAB connection allows the development of a new controller prototype. This prototype is transformed into a controller model with the use of the LIBCREG library. This model runs about twice as fast and is independent of MATLAB. At a technical level, an efficient sensibility (simulation-based parameter optimization) is easily performed. In particular, the controller parameters are introduced via the CLIM 2000 interface and a new simulation is run which is automatically documented and archived using 'catapost'. If the developer chooses to make a change to the controller, he can do so without affecting the integrity of the simulation. That is, post-treatment of the simulation data is still possible via CLIM 2000's standard process.

REFERENCES

[1] Bonneau, D., Rongere, F.X., Covalet, D., and Gautier, B., (1993). CLIM2000: Modular software for energy simulation in buildings. IBPSA 93, Adelaïde, August.

[2] Remond, B., Déqué, F., (1995). *Connexion CLIM 2000 MATLAB: Etude de faisabilité*. Moret-sur-Loing, France: Les Renardières, ADEB, HE-14/95/046, 1995.

[3] Murphy, K.M, Rémond B., Déqué F, Controller design for climate control in building using CLIM 2000, CLIMA 2000, Liege, 1997.

[4] Murphy, K. M., Déqué, F. (1996). Generic Controller Simulation Interface Library for CLIM 2000; Feasibility Study, Moret-sur-Loing, France: Les Renardières, ADEB, HE 14/96/045, 1996.

[5] Stangerup, P. ESACAP User's Manual, StanSim: Denmark, April 1994.

[6] MATLAB External Interface Guide (UNIX), The Mathworks Inc., Jan. 1993.

[7] Boucher, P., Dumur, D., Murphy K.M, Déqué F. On predictive Controller design for comfort control in single residential housing. ECC 97, Brussels, july 97