



Exploring the use of traditional heat transfer functions for energy simulation of buildings using discrete events and quantized-state-based integration

Víctor-Manuel Soto-Francés, Emilio-José Sarabia-Escrivá, José-Manuel Pinazo-Ojer & Pedro-Juan Martínez-Beltrán

To cite this article: Víctor-Manuel Soto-Francés, Emilio-José Sarabia-Escrivá, José-Manuel Pinazo-Ojer & Pedro-Juan Martínez-Beltrán (2020): Exploring the use of traditional heat transfer functions for energy simulation of buildings using discrete events and quantized-state-based integration, Journal of Building Performance Simulation, DOI: [10.1080/19401493.2020.1723704](https://doi.org/10.1080/19401493.2020.1723704)

To link to this article: <https://doi.org/10.1080/19401493.2020.1723704>



Published online: 06 Feb 2020.



Submit your article to this journal [↗](#)



Article views: 10







View related articles [↗](#)



View Crossmark data [↗](#)



Exploring the use of traditional heat transfer functions for energy simulation of buildings using discrete events and quantized-state-based integration

Víctor-Manuel Soto-Francés ^a, Emilio-José Sarabia-Escrivá ^a, José-Manuel Pinazo-Ojer ^a and Pedro-Juan Martínez-Beltrán ^b

^aDepartamento de Termodinámica Aplicada, ETSII, Universitat Politècnica de València, C/Camino de Vera s/n, Valencia 46022, Spain; ^bDepartamento de Ingeniería Mecánica y Energía, Elche, Universidad Miguel Hernandez de Elche, Alicante ES 03206, Spain

ABSTRACT

The target of the paper is to study how to devise an efficient discrete-event model for the yearly energy simulation of buildings. Conventionally, software tools use time-driven schemes and many components must be computed at every sampling time-point. Event-driven simulation aims at lowering this burden, by calling only those components whose state is evolving quickly. The article explores a model based on DEVS formalism and Quantized State Systems (QSS) techniques. Within this paradigm shift, our strategy was to reuse as much widely accepted knowledge as possible. One immediate difficulty was, that the well-known conduction heat transfer function (CHTF) of multi-layered walls is not suitable for DEVS in its traditional form since it is constrained to sample at a fixed time step. Instead, the paper introduces a non-conventional method: the *Successive State Transition* method (SST). Its distinguishing traits are: it allows variable time steps, has high accuracy and its computational workload adapts to the elapsed time between transitions. Unfortunately, although we found that SST and QSS work well together, the paper shows that the aforementioned transfer function is not adequate for event-driven simulations. Based on the paper outcomes, we propose a workaround for further research: a new transfer function, relating the conduction heat flux (input) to the *time derivative* of the wall superficial temperature (output) (recall that the traditional input-output relationship: superficial temperature and conduction heat flux, respectively).

Abbreviations: CHTF: Conduction Heat Transfer Functions; DEVS: Discrete Event Simulation; DRFM: Direct Root Finding methods; SST: Successive State Transition method; QSS: Quantized State System

ARTICLE HISTORY

Received 15 February 2018
Accepted 21 January 2020

KEYWORDS

Successive state transition method; heat transfer; discrete event simulation; DEVS; buildings; energy simulation

1. Introduction

Nowadays the increasing computation power and low prices of personal computers is leading to a new paradigm in calculation methods applied to complex engineering systems. In this context, energy simulation of buildings is a growing field (see Li et al. 2015).

Any building as a *technological object* is, in fact, a very complex system where, in our opinion, the Discrete Event System Specification (DEVS) formalism suits perfectly (see Ziegler 2000). DEVS can be used for developing either discrete-event or discrete-time simulations. The discrete-event method focuses on state events issued by any simulation component at any time while in *discrete-time*, the time step is the only event or driving force of the calculation. Therefore very non-linear phenomena, like ON/OFF controls, or random signals like occupancy (see Gunay et al. 2014) fit naturally.

In industry, this idea of non-monolithic simulation, is materialized with the Functional Mock-up Interface standard FMI which provides the means for model based development of complex systems. Building energy simulation tools like EnergyPlus are already scheduling a change into this new paradigm (see SOEP; Wetter and Noudui 2015) and the IEA (International Energy Agency) IEA-annex60 also supports a model based development.

1.1. Previous works on quantized state integration applied to build energy modelling

To our knowledge, there have been few attempts to use discrete-event methods in building energy calculations (see for instance Zimmermann 2001). More recently the work by Goldstein, Breslav, and Khan (2013) shows how DEVS, from a mathematical perspective, accommodates the co-simulation strategies known as loose and strong coupling, as well as, strategies involving variable time steps. A recent work by Bergero et al. (2017) uses a very specialized engine to integrate quantized state systems (QSS). Roughly, a QSS is a conventional dynamical system whose variables may only change in chunks called quanta. They studied very simplified rooms and air conditioning systems. Their room model is of the thermal-network type. Unfortunately, the infra-red heat exchange inside the rooms was not considered. Previously Gunay et al. (2013) studied a similar type of thermal-network model but for a single room, modelling the infra-red heat exchange as constant thermal resistances among the surfaces (i.e. linearizing the radiation heat exchange). Their objective was to see the effect of random loads or excitations like the occupancy. They applied a quantization integration scheme, without a concrete specification, where the state dynamics of every node was tracked.

The heat conduction calculation method lies at the core of any building energy simulation and is still a controversial question. Recently Mazzarella and Pasini (2015) published a discussion about this. Incidentally, we would add the differential-difference scheme (see Pakanen 1996) as an intermediate method, between the two big Mazzarella's classification sets; CHTF (Conduction Heat Transfer Function) and finite differences. Mazzarella splits CHTF into: the SS (State Space), Direct Root Finding and Frequency Domain Regression (FDR) subsets. His main outcome is that CHTF methods are quicker for big time steps (1[h]), but for smaller steps (15[min]), finite differences and CHTF are similar. Additionally, according to EnergyPlus (2012) Section 3.1.3 page 62 (version 9.0.1): '... conduction transfer function series become progressively more unstable as the time step decreases ... related to round-off and truncation error ... finite difference approximations were considered ... to address this problem'. Other authors (see Maestre, Cubillas, and Pérez-Lombard 2010) are in favour of Laplace methods because of its accuracy and speed. CHTF methods use a hold function between sampling time-points. The surface temperature profiles are shaped by this externally imposed hold function thus allowing greater time advances. The traditional hold function, is the linear one.

However, this linearity and the selection of the temperature as the input signal comes with a hidden cost. As Ojer et al. (2015) showed, in a simple case, the penalty of choosing large time steps along with the linear hold function is an unbalance, within the time step, of the energy (in form of radiation, convection and conduction) that crossed the external surfaces of the multi-layered slab. This shows up as a fictitious increase of the wall heat capacity or thermal inertia. Perhaps this has something to do with Li et al. (2015) comments at their conclusions section; 'more research is needed because the simulation tools tend to underestimate energy consumption'. Also, although not so clearly stated, Ko and No (2015) showed discrepancies between simulated and measured values.

Here, we present a very basic multi-zone building model for discrete-event simulations. It extends previous work by Francés, Escrivá, and Ojer (2014), for a single room. We have used the conventional conduction heat transfer function of multi-layered slabs (CHTF) which relates surface temperatures to heat fluxes (see Equation (A2)). However, in order to be used by a discrete-event engine, the successive state transition method (SST) has been employed (see Appendix 1 or Francés, Escrivá, and Ojer 2014 for details). The SST method also uses a hold function, but allows a variable time step (see Francés, Escrivá, and Ojer 2015; Ojer et al. 2015). The SST could be considered to belong to the set of Direct Root Finding methods. If short-time responses are needed the SST has a good workload self-adjusting capability (see Section 2.1 and Appendix 1 and see Ojer et al. 2015). Our proposal keeps a key idea behind traditional CHTF methods: solve the multi-layered 1D conduction heat transfer beforehand, leaving as explicit state variables, only those of the room/zone air. Our intention was twofold: large component elapsed times and *local* reaction to events, thus increasing the SST speed and avoiding calling concurrently, all the conduction elements. Unfortunately, as it will be argued, the current implementation has problems which did not come up in the single-room case.

However, the positive aspect is that it hints at a possible solution.

Previously Francés, Escrivá, and Ojer (in 2014), we only tested the QSS integrator for quantized systems, while here we include variants named centred (CQSS) and linearly implicit (LIQSS) integrators.

Finally, building simulations require dry air and water mass balances, energy balances and air flow-networks. Traditionally, calculations are done in separate and specialized realms (i.e. energy, flow-networks, HVAC systems, etc.) and some kind of linking is devised among them (see Wang and Zhai 2016 or Trcka, Hensenaand, and Wetter 2009). Our ultimate goal is to create a discrete-event model which includes all of the phenomena.

2. Discrete-event building energy model

This section presents key aspects of the discrete-event building model.

2.1. Successive state transition method

Figure 1 shows the multi-layered conduction element along with the scheme of the transfer functions. By transfer functions, we understand Laplace's transform function relating an input to an output. In building energy simulation, the traditional transfer functions take as excitations, the temperatures at either side of the slab/wall and, as outputs, the conduction heat fluxes. By G_{01} it is meant the excitation at *side* = 0 and the response at *side* = 1. Briefly, the idea is that the model receives trains of temperature pulses at either side, unevenly located in time and the (linear) hold function shapes the temperature evolution between two consecutive pulses. The conduction heat flux is the output signal and it is sampled at the same time-points as the input pulses. In other words, after a temperature input event at either side, the method outputs the conduction heat flux at each side assuming that the surface temperature evolved linearly from the last known temperature at that side to its current value. Notice that since the wall works as a single element, this temperature change at one side induces an evaluation of the temperature at the other side¹. The new output conduction heat fluxes are evaluated and the state of the wall is updated.

Explaining the details of SST is out of the scope of the paper. A very detailed explanation of the use of SST can be found in Francés, Escrivá, and Ojer (2014). For completeness, the basics of the Successive State Transition method can be found in Appendix 1.

However, we provide here a summary of the key ideas. Roughly, the wall model comes from partial differential equations. Therefore the model has an infinite amount of *relaxation modes* which characterize (along with other information) the wall response. The characteristic response time of each mode is obtained from the roots of a function and are negative numbers named poles (that is the origin of being called a Direct Root Finding method). The more negative, the quicker the response decays. The user of the SST chooses the maximum amount of poles to be used during the simulation, according to the expected shortest excitation time. After an input signal (surface temperature change), according to the elapsed time t_e from the last input change, a variable amount of modes are excited. These

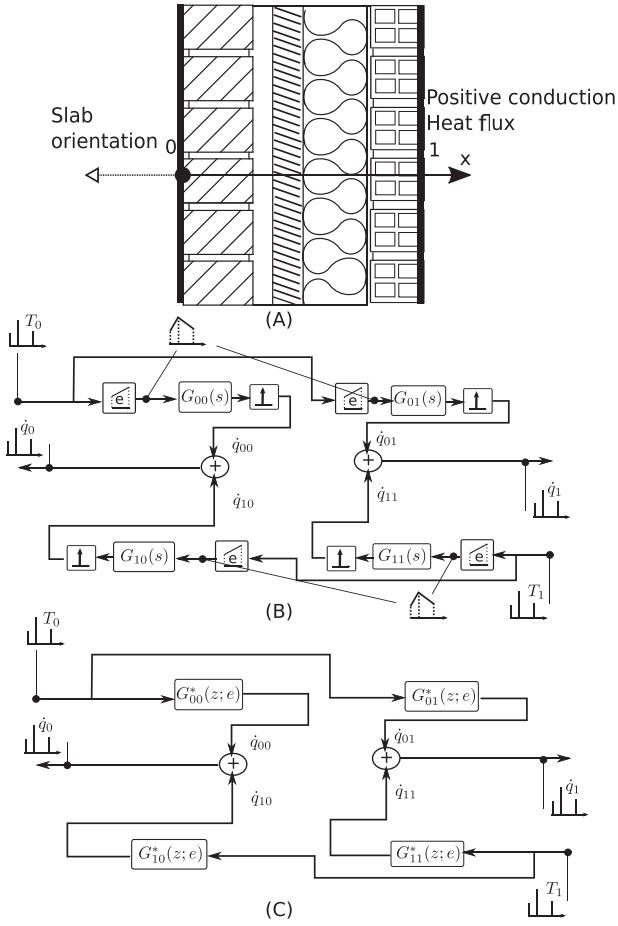


Figure 1. Transfer functions scheme for a multi-layered slab. (A) Reference axis. (B) Details of the hold/shaping function parametrized by the elapsed time t_e and the output sampler. (C) Simplified scheme in Z-transform form.

modes are needed to build up the output (i.e. conduction heat flux). We stress that not all the available modes need to be computed. This information is stored as an internal state while the wall element waits for another input. At the next input event, if t_e is large then many of the modes have already disappeared and many states can be simply discarded. In this way, the transition of the state of the wall is successively computed in time. In summary, SST has two great advantages: allows a variable time step and the workload of computing both, the state transition and the response, goes down with t_e .

2.2. Building energy balance model

In Francés, Escrivá, and Ojer (2016) we proposed an ordinary differential equations system to model the multi-zone simultaneous energy, air and water mass balance. It was devised bearing in mind its future use in discrete-event simulations. Nevertheless, here we focus exclusively on solving the energy balance inside the rooms. Thus, there is no inter-flow of water or dry air among the rooms or, in other words, each room is airtight with a constant amount of dry air and no ventilation. The energy balance just expresses that the sum of the convective heat fluxes from each boundary surface into the room air, plus the internal convective sources inside the room, equals the energy storage rate into the room air.

Firstly, let us briefly see, the general pose of the problem without any discrete-event consideration. For the n -room, the energy balance is written as:

$$\dot{E}_n(t) = m_n \cdot c_{da} \cdot \dot{T}_n = \sum_{j \in B_n} [\dot{Q}_{conv,j[side]}(t)] + \dot{Q}_{src-conv,n}(t), \quad (1)$$

where B_n is a subset of J (the set of all the conduction elements) such that $B_n = \{j \in J | j \text{ is a boundary of room } n\}$, where,

$$\dot{Q}_{conv,j[side]}(t) = h_{j[side]} \cdot A_j \cdot (T_{j[side]}(t) - T_n(t)) \quad (2)$$

this dependency can be just written as:

$$\dot{T}_n(t) = f_n(\dots, \dot{Q}_{conv,j[side]}(t), \dots, \dot{Q}_{src-conv,n}(t)) \quad j \in B_n. \quad (3)$$

The convective signal $\dot{Q}_{conv,j[side]}$ can be evaluated continuously in time by using the traditional heat balance method. This method consists of the superficial heat power balance at each $side \in \{0, 1\}$ of the j -boundary element (see Equations (4) and Figure 2). The unknowns are the surface temperatures $T_{j[side]}$ which keep the balance. As it was explained in Section 2.1, the wall has internal states related with the previous excitations, due to changes in its surface temperatures. These *excited* states decay exponentially with the elapsed time from the last excitation (see Appendix 1). Those temperatures are used in Equation (2) to get $\dot{Q}_{conv,j[side]}$.

The power balance method is illustrated in Figure 2 and represented mathematically by Equations (4).

$$\dot{Q}_{conv,j[0]}(t) + \dot{Q}_{rad-lw,j[0]}(t) + \dot{Q}_{cond,j[0]}(t) = \dot{Q}_{src,j[0]}(t), \quad (4)$$

$$\dot{Q}_{conv,j[1]}(t) + \dot{Q}_{rad-lw,j[1]}(t) = \dot{Q}_{cond,j[1]}(t) + \dot{Q}_{src,j[1]}(t).$$

The infra-red heat exchange has an algebraic dependency on the surface temperatures as Equation (5) shows.

$$\begin{bmatrix} Q_{rad-lw,1}/A_1 \\ Q_{rad-lw,2}/A_2 \\ \dots \\ Q_{rad-lw,Bn}/A_{Bn} \end{bmatrix} = \begin{bmatrix} \frac{1}{\epsilon_1} - F_{1,1} \frac{(1-\epsilon_1)}{\epsilon_1} & \dots & -F_{1,Bn} \frac{(1-\epsilon_{Bn})}{\epsilon_{Bn}} \\ -F_{2,1} \frac{(1-\epsilon_1)}{\epsilon_1} & \dots & -F_{2,Bn} \frac{(1-\epsilon_{Bn})}{\epsilon_{Bn}} \\ \dots & \dots & \dots \\ -F_{Bn,1} \frac{(1-\epsilon_1)}{\epsilon_1} & \dots & \frac{1}{\epsilon_{Bn}} - F_{Bn,Bn} \frac{(1-\epsilon_{Bn})}{\epsilon_{Bn}} \end{bmatrix}^{-1} \cdot \begin{bmatrix} 1 - F_{1,1} & \dots & -F_{1,Bn} \\ -F_{1,2} & \dots & -F_{2,Bn} \\ \dots & \dots & \dots \\ -F_{Bn,1} & \dots & 1 - F_{Bn,Bn} \end{bmatrix} \cdot \begin{bmatrix} \sigma \cdot T_{1,[side]}^4(t) \\ \sigma \cdot T_{2,[side]}^4(t) \\ \dots \\ \sigma \cdot T_{Bn,[side]}^4(t) \end{bmatrix}. \quad (5)$$

Finally, the conductive heat flux signal $\dot{Q}_{cond,j[side]}$ can be solved as shown in Section 2.1.

Traditionally, when using discrete-time methods, one may imagine the situation as if there was an imaginary clock, that issues events at a certain rate, the well-known time step. Usually, nobody thinks of it as an actual component of the system of equations, but as an element of the integration scheme. At any sampling instant t , Equation (4) is evaluated for each j -element

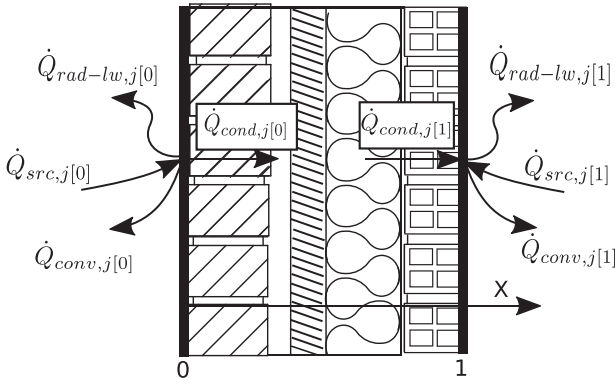


Figure 2. Balance method at conductive j -element. Superficial thermal power balance at each side of a multi-layered conductive element.

and the output is the set $\{T_{j[side]}\}$ of all the surface temperatures along with the set $\{\dot{Q}_{conv,j[side]}\}$. Therefore, the variables $\{T_{j[side]}\}$ are not continuous any more, but concurrent and discrete in time. Between sampling instants, due to the lack of information, a hold function is employed to shape the evolution profile of the surface temperatures $\{T_{j[side]}\}$. It should be stressed that the profiles are imposed as external input excitations to the multi-layered heat conductive elements. Traditionally, this hold function is a triangle that creates a linear interpolation between the last and the new surface temperature values. Notice that this is just an arbitrary choice (see Francés, Escriva, and Ojer 2015; Ojer et al. 2015) and may even not correspond to an actual physical evolution. As mentioned, this profiling acts as an internal mechanism for handling, beforehand, the heat conduction problem. Its single output signal, in the linear case (see Ojer et al. 2015 for other possibilities), is the conduction heat $\dot{Q}_{cond,j[side]}$ at the sampling event at t . Finally, this output value is kept constant between sampling instants due to the lack of information and, therefore, up to the first order of approximation, the same is assumed for the rest of the heat fluxes (like the convective or the radiative infra-red heat flux).

Usually, the discrete-time methods use a predefined and fixed time step which acts, in fact, as a simulation parameter not as a variable. It could be said that the functions f_n in the right hand side of (3) are evaluated according to the pulses from a clock.

Summarizing, conventional discrete-time methods think of the building dynamics, as follows:

$$\dot{T}_1 = f_1(T_1, \{T_{j_1[side]}\}, \dot{Q}_{src-conv,1}), \quad j_1 \in B_1$$

$$\dot{T}_2 = f_2(T_2, \{T_{j_2[side]}\}, \dot{Q}_{src-conv,2}), \quad j_2 \in B_2$$

...

$$\dot{T}_N = f_N(T_N, \{T_{j_N[side]}\}, \dot{Q}_{src-conv,N}), \quad j_N \in B_N$$

Additionally at every time step t_m the following equations must be solved :

- * Infrared radiant heat exchange in each i -room,
 $\rightarrow \{\dot{Q}_{rad-lw,j[side]}\}$

- * Balance method equation at each surface $j[side]$,
 $\rightarrow \{T_{j[side]}\}$
- * Heat flux conduction at each j -element
 $\{\rightarrow \dot{Q}_{cond,j[side]}\}$
- * Convective heat flux at each surface, $\rightarrow \{\dot{Q}_{conv,j[side]}\}$
- * Room convective model $\rightarrow \{h_{conv,j[side]}\}$
- * $\dot{Q}_{src}(t)$, (for instance, weather signals or occupancy).
- * Clock , Δt parameter, $t_{m+1} = t_m + \Delta t$ (6)

In the Equations (6) the arguments of any function f_n show explicitly the functional dependency on the room air temperature T_n , the internal convective sources $\dot{Q}_{src-conv,n}$ and the boundary surface temperatures $\{T_{j_n[side]}\}$ at each time step t_m . Notice that in turn the $\{T_{j_n[side]}\}$ depend, in fact, on all the other room air temperatures and on the external sources but they do not appear as dynamical variables in Equation (6). Therefore the f_n could have been written formally as:

$$\dot{T}_n = f_n(T_1, \dots, T_N, \dot{Q}_{src-conv,1}) \quad (7)$$

Remark: Why these arguments have been chosen explicitly will be clarified ahead, when dealing with the DEVS model.

The algebraic equations below the differential equations in (6) are evaluated at each time event triggered by the clock in (6). Usually, the other input or source signals \dot{Q}_{src} are also evaluated at sampling points and remain constant within them. Notice that the set of differential equations does not depend explicitly on time and thus the system is autonomous. Finally, to integrate the differential equation system (6) the literature uses different well-known quadrature schemes.

2.3. Discrete-event version of the building

Our approach must be necessarily different since we need a general event-driven (not a time-driven) evaluation of the right hand side terms f_n of equations (6). At any time, many different types of events can trigger the evaluation of the superficial balance equations (4) and hence of the f_n terms. For instance to name a few: the change in the weather conditions (solar radiation, outdoor temperature, etc.), a change in any of the room air temperatures, an internal gain, a control and so on. The successive state transition method was chosen because it allows to use a variable time step along with a *hold function*. This means that now the new $\dot{Q}_{cond,j[side]}$ at both sides are determined by both, the new $T_{j[side]}$ and the elapsed time t_{ej} since the last input event to the j -conductive element. In other words, the same surface temperature change but with a different elapsed time would output different $\dot{Q}_{cond,j[side]}$ values. This implies that the same time dependency passes on to the convective heat $\dot{Q}_{conv,j[side]}$ and to the f_n functions.

The conventional problem (6) turns now into an event-driven system. All the algebraic equations remain the same but the last two since now the events come from input sources distributed

unevenly in time:

$$\left. \begin{array}{l} * \dot{Q}_{src}(t) \text{ weather, occupancy, etc.} \\ * \text{Clock } , \Delta t \text{ parameter, } t_{m+1} = t_m + \Delta t \end{array} \right\} \rightarrow \dot{Q}_{src} \text{ random weather, occupancy, etc.} \quad (8)$$

Now a new difficulty arises. The system (6) becomes non-autonomous, i.e. depends explicitly on time but in a subtle way. More exactly, although all the f_n arguments were constant, as time goes by, the time derivative of the n -room air temperature would change due to the introduction of the time dependency of the heat fluxes of the walls (concretely with t_e). Expressed in another way Equation (6) now becomes: $\dot{T}_n = f_1(T_1, \dots, T_N, \dot{Q}_{src-conv,1}, t)$. Moreover, now neither the set of surface temperatures $\{T_{j[side]}\}$ will be concurrent, nor their computed amount will be the same for every j -element. It will depend on how quickly events happen around that j -element. The system is just driven by the weather events signals or other source signals (like the internal gains). Sources may issue events to some subset of the boundary conductive elements (like the external walls, in case of the weather). Moreover, diverse causes may issue events; schedules, random signals or even rule-based events (for instance, a change by a certain amount of solar radiation) and so on.

The new system is not ready yet for discrete-event simulation. We need to see the changes of the air temperature of the rooms. However, these temperatures are present in a continuous differential equations set. Fortunately, there exists a solution method which allows to integrate them by using events. The idea is based on the quantized state system method (see Migoni 2010). The set of state variables $\{T_n\}$ is replaced by its quantized counterpart $\{q_{T_n}\}$. Roughly explained, an event is triggered whenever any quantized variable crosses a predefined threshold, defined by a quantum (for instance, a temperature change of $0.1[^\circ\text{C}]$). The numerical integration of quantized systems is known generically in the literature as QSS methods and there are different schemes (named for instance QSS, CQSS, LIQSS, BQSS) some with several orders of accuracy (QSS1 for first order, etc.). In Appendix 5 we explain briefly the idea of QSS methods, but for very detailed explanations the reader is referred to Migoni (2010).

Some final remarks should be done. The differential equations system can be written in a more canonical form (see Equation (9)) by using, as before, the dependency of $\{T_{j[side]}\}$ on the quantized variables q_{T_n} .

$$\left. \begin{array}{l} \dot{T}_1 = f_1(q_{T_1}, \dots, q_{T_N}, \dot{Q}_{src-conv,1}, t) \\ \dot{T}_2 = f_2(q_{T_1}, \dots, q_{T_N}, \dot{Q}_{src-conv,2}, t) \\ \dots \\ \dot{T}_n = f_n(q_{T_1}, \dots, q_{T_N}, \dot{Q}_{src-conv,n}, t) \end{array} \right\} \quad (9)$$

Moreover, the differential equations could have been written in the energy content of the room air, instead of its temperature (see Eq. (10)). In this case, the *quantum* would be an energy content change of the room air. Notice that the same quantum is applied to all the rooms. Therefore, in a general case, depending on the size of the room, the change of temperature that triggers

an event would be different.

$$\left. \begin{array}{l} \dot{E}_1 = f_1(q_{E_1}, \dots, q_{E_N}, \dot{Q}_{src-conv,1}, t) \\ \dot{E}_2 = f_2(q_{E_1}, \dots, q_{E_N}, \dot{Q}_{src-conv,2}, t) \\ \dots \\ \dot{E}_n = f_n(q_{E_1}, \dots, q_{E_N}, \dot{Q}_{src-conv,n}, t) \end{array} \right\} \quad (10)$$

This last option was chosen here. As a general rule, it is preferred since it should track more accurately, the energy flows.

2.4. Implementation

Each particular software implementation of a DEVS simulation engine may exhibit different features. We have used the implementation by Bergero and Kofman (2011) named PowerDEVS. Therefore the exposition of our implementation follows the implementation constraints imposed by the PowerDEVS engine.

Figure 3 shows the building model to be defined as a DEVS model. The model has not complexities like windows or ventilation and it is not coupled to the ground since our intention is to illustrate the basics and to search for the foundations of a good performance discrete-event simulation method. In order to study several building configurations, the model is built upon a modular approach (see Figure 3). Every room has the same size and all the rooms are arranged matrix-wise so that by the notation $a \times b \times c$ we mean a building with a , b and c rooms in x , y and z directions respectively.

In DEVS there are two types of components; atomic and coupled. Any coupled component represents a sub-system made up of atomic and possibly other coupled components. The DEVS model to be simulated is in itself a coupled model. PowerDEVS uses a positive integer as a component identifier *ID* (starting at $ID = 0$). The *ID* increases in the same order in which the component is declared inside the coupled element to which it belongs. This order is important since it acts as the Select function of the DEVS formalism. The role of this function is to select, in case of concurrency among several components, the order in which they are called. Concurrency appears naturally when there is the need to solve recursively a system of algebraic equations during the simulation. Therefore in PowerDEVS, in case of concurrency, the concurrent components are called in the same order in which they are declared. Each component has input and output ports identified by positive integer numbers. Every coupled component is defined by two lists; one for the components and another for the port connections (also called structure), among them. An atomic component is defined by its type and by a list of arguments that define its non-structure dependent properties (for instance, for a wall-component, its composition, orientation, area and so on). Figure 4 shows a DEVS model scheme for a $(2 \times 1 \times 1)$ -building. Each box in the figure represents a component of a certain type. The numbers on the left and right are input and output ports respectively. As aforementioned, the position of each component in this figure is not arbitrary. The components are declared in the PowerDEVS input file, according to the following scheme; first by moving from top to bottom and then from left to right. In Figure 4 this rule is partially broken for the WALLS component list for a better visualization. However, in this particular case, the order of the walls does not matter since they

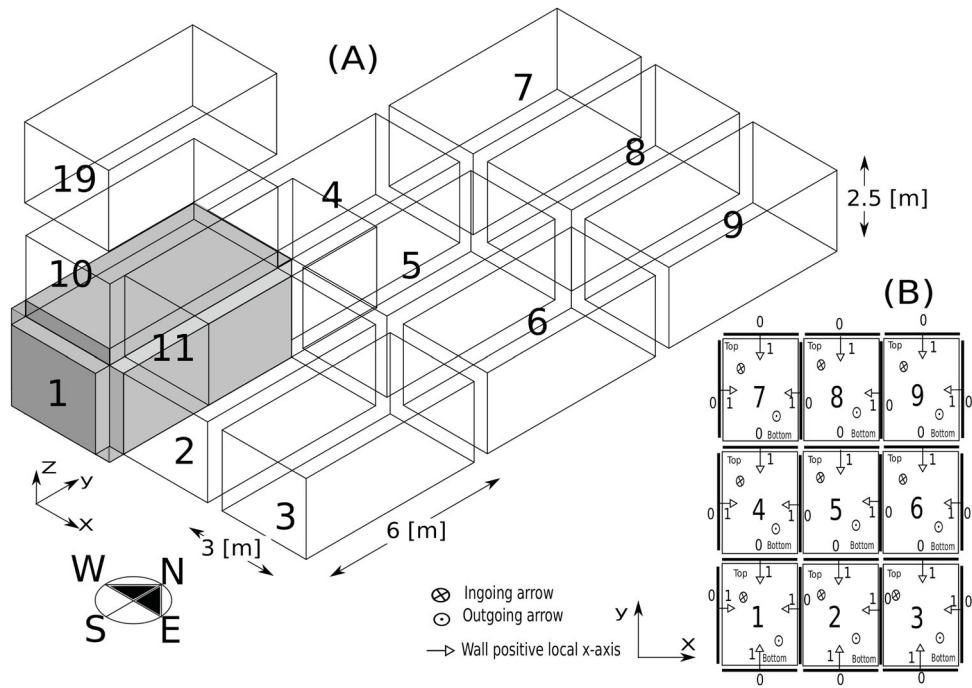


Figure 3. (A) $(3 \times 3 \times 3)$ -building model, the numbers indicate the order in which the rooms are declared in the DEVS model. (B) 1st-floor top view: shows the senses and numbering of every side of the conductive elements.

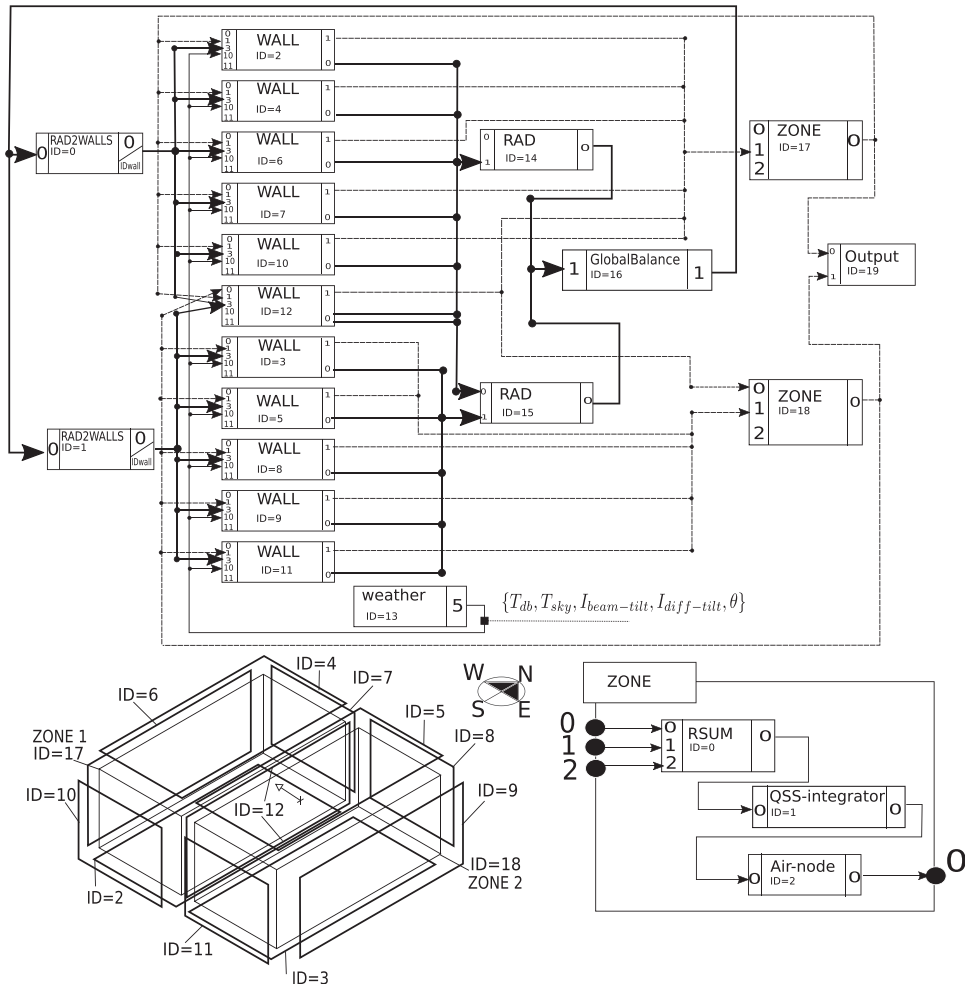


Figure 4. Scheme of the DEVS model of a $(2 \times 1 \times 1)$ -building, that shows the details of the connections.

can be called in any order (i.e. inside their own column in the scheme).

3. Discrete-event model explanation

This section presents the multi-zone model and explains its design.

3.1. Model component types

As the example of Figure 4 shows our DEVS model proposal uses one coupled and nine types of atomic components. The role of each component is summarized as follows:

Note: By WALL we refer to any multi-layered slab whose conduction heat must be computed.

- (1) RAD2WALLS: This component is in charge of calling a subset of the WALL set during an iteration. It could have been implemented just with a single component but for implementation reasons, there is one component per room.
- (2) WALL: This simulates any multi-layered conductive j -element and forces the local superficial heat flux balance (Equation (4)). Its outputs are the $T_{j[side]}$ and the $\dot{Q}_{conv,j[side]}$. If the elapsed time, after an external event, is not zero then it computes its *state transition*, i.e. computes the conduction heat flux coefficients $\{a_{00}, a_{01}, a_{10}, a_{11}, D_0, D_1\}$, using Equations (A6b) and (A7j). Regardless of the elapsed time and during an infra-red heat exchange iteration, it computes the superficial heat flux balances at both sides, clearing for the new temperatures $T_{j[side]}$, $side \in \{0, 1\}$ which fulfil such balance with the current $q_{rad-lw,j[side]}$. The converged new state is computed with Equation (A8) using the converged temperatures.
- (3) WEATHER: In our implementation, it is a pure forcing function (i.e. has no input-ports). It issues an event every hour. Nevertheless any other rule could be employed (like splitting the signals into events due to partial changes in; solar radiation, sky temperature, dry-bulb and wet-bulb temperature).
- (4) RAD: Computes the infra-red radiation heat exchange among the boundary surfaces of a room using the $T_{j[side],side \in \{0,1\}}$ (see Equation (5)). There are as many RAD elements as rooms. The view factors of our modular room are shown in Table 1. The way to sort the room boundary surfaces in order to compute the view factors among them so that RAD is able to assign correctly those values, is to form a six-tuple (since we have six walled rooms) as follows: (step 1) if the boundary surface of the ZONE corresponds to the $side = 0$ of the WALL the *ID* assigned to a WALL-component

is taken with positive sign and negative otherwise, (step 2) the output set of numbers from step 1 is sorted from the lowest to the highest value. For instance, in Figure 4 the results are: ZONE1 :(-12, -10, -7, -6, -4, -2) and ZONE2 :(-11, -9, -8, -5, -3, 12). (Remark: Recall Figure 3 about the general rule used to orient the WALLs).

- (5) GLOBALBALANCE: This component controls the superficial heat power balance fulfilment for all the building WALLs. In fact acts as a control for the concurrent iteration over the infra-red heat exchange.
- (6) ZONE (coupled DEVS element): It represents the conservation laws applied to the room (just energy in our case).
 - (a) RSUM: This element adds the convective power \dot{Q}_{conv} from any origin. In other words, it computes the right hand side of Equation (1).
 - (b) QSS-integrator: It is a (first order) quantized state integrator of any type $\{QSS1, CQSS1, LIQSS1, etcetera\}$.
 - (c) Air-node: Our original approach was to use the differential equations in energy form (see Eq.(10)). Therefore this component transforms the energy content of the room air into its dry-bulb temperature (see also Francés, Escrivá, and Ojer 2014).

3.2. Overview and features of the proposed DEVS building model

Showing the details of each component falls out of the extent of the paper. A summary of key points can be found in Appendix 4. For a better understanding we prefer to expose an overview about how all the components work together.

The main guiding ideas in developing an efficient model should be; avoiding high frequency events (oscillations not due to the model dynamics) and within any event to reduce the amount of concurrent element calls. A good building model would not try to call all the conductive boundary elements of the building (all the rooms), after an event.

The main difficulty to achieve that is how to extend the tracking of the superficial heat power balance inside a room between events without the need of computing the state transition since it is a computationally expensive task (see Equations (A6b) and (A7j)). Additionally the infra-red radiation heat exchange acts instantaneously inside the rooms thus creating thermal couplings that break the superficial heat power balance (see Equations (4)) among the WALLs. Therefore, at first sight it seems that there would be a concurrent need for a state transition propagation through all the building conductive elements after a local event.

To deal with this problem, we have addressed two sub-problems:

Table 1. View factors of the room module.

Number	Floor	Roof/ceiling	West Wall	North wall	East wall	South wall
Floor	0.00000	0.34276	0.22224	0.106380	0.22224	0.106380
Roof/ceiling	0.34276	0.00000	0.22224	0.106380	0.22224	0.106380
West Wall	0.26668	0.26668	0.00000	0.107750	0.25114	0.107750
North wall	0.25532	0.25532	0.21550	0.000000	0.21550	0.058366
East wall	0.26668	0.26668	0.25114	0.107750	0.000000	0.107750
South wall	0.25532	0.25532	0.21550	0.058366	0.21550	0.000000

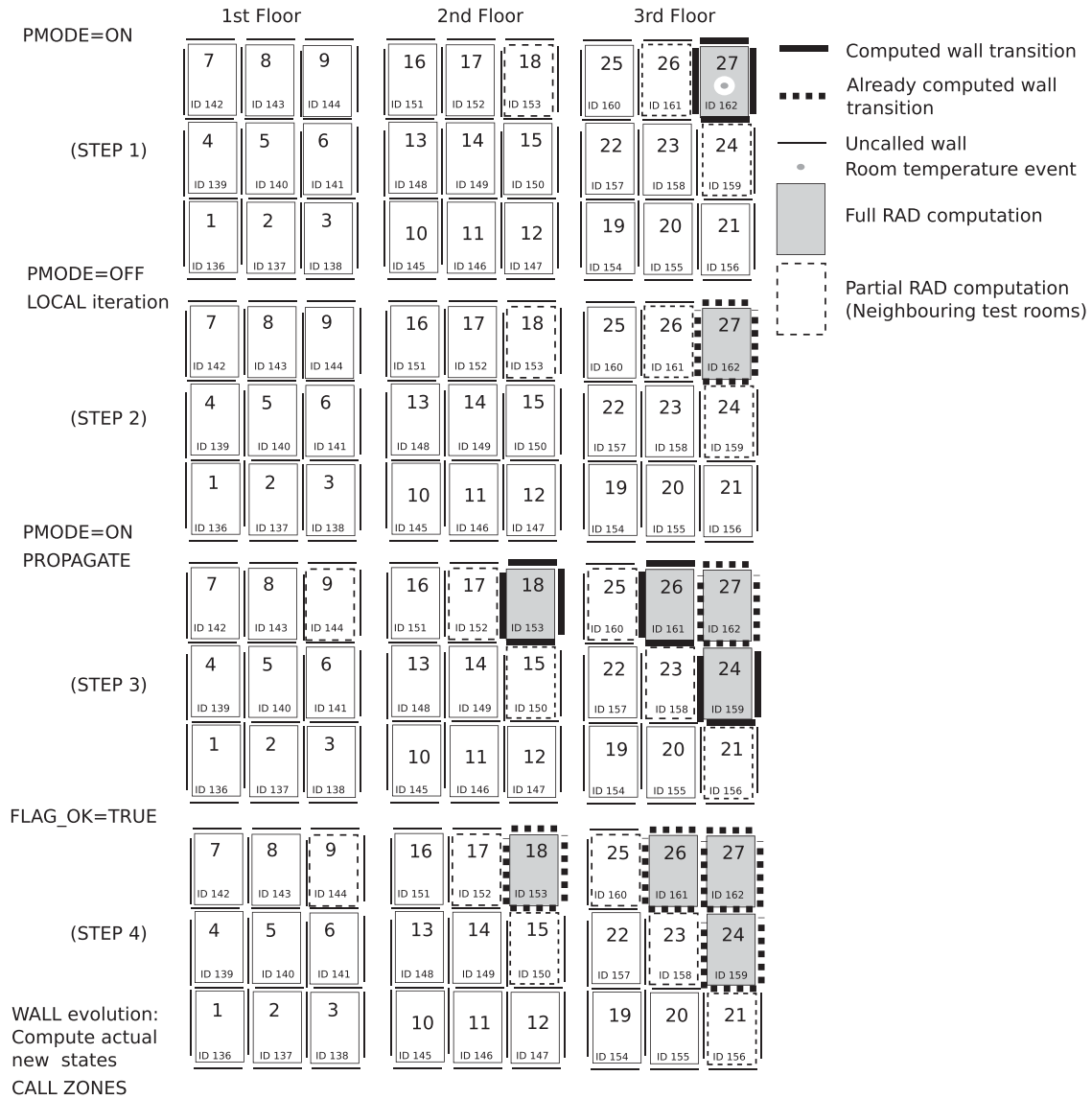


Figure 5. $3 \times 3 \times 3$ -Building. Example of concurrent propagation of wall evolution.

- The WALLs thermal relaxation: after an elapsed time t_e due to an incoming event, the superficial temperature of previously excited walls might still be changing rapidly. This should be checked by the model.
- The thermal coupling induced by the infra-red heat exchange: a change in $q_{rad-lw,j[side]}$ at just a single side of a wall, changes the temperatures at both sides. In turn this makes changes in $q_{rad-lw,j[side]}$ at the rest of the walls of the same room but also at the other side thus affecting the neighbouring rooms and propagating the need for the conduction state transition calculation.

The following exposition is based on Figure 5 that represents a three-floored building made up of 27 rooms. Additionally a scheme like the one in Figure 4, but a bit more complex for this $3 \times 3 \times 3$ -building must be imagined. The steps are used to understand the message processing after an event. We stress that by 'propagation front' it is meant those elements which are going to be called for a state transition *concurrently* due to any event.

step (1) Imagine that ZONE 27 emits a temperature change event after a time advance t_a from the last simulation call. All their boundary WALLs receive that message and their state transitions (after an elapsed time t_e , which may be different for each WALL) are computed. This breaks the heat power balances Equation (4), at the WALLs surfaces facing ZONE 27. The new surface temperatures of those walls, which restore the balance, must be computed. That means that, simultaneously, the WALLs surfaces facing the neighbouring rooms have also their heat power balance equations broken (ZONES 18, 24, 26). Therefore, each WALL element computes new temperatures at both sides. In doing so, the previous values $q_{rad-lw,j[side]}^{old}$ are used. Concurrently, the new surface temperatures are sent to the corresponding RAD elements. The RAD for ZONE 27 will solve an unconstrained equation system like (5) clearing for the q_{rad-lw}^{new} . However, for the neighbouring zones, the temperatures and states of the non-called WALLs must be kept fixed during the RAD call for the q_{rad-lw}^{new} (see

Appendix 2). The change in $q_{rad,lw}$ is measured as:

$$\begin{aligned} & Error_{rad-lw} \\ &= \max_{\text{called RADs}} \{|q_{rad-lw,j[side]}^{new} - q_{rad-lw,j[side]}^{old}|\} \\ &< TOL_{rad-lw} \end{aligned} \quad (11)$$

Note: notice that the max function is extended only to all the RADs of previously called rooms.

The GLOBALBALANCE component is in charge of controlling the iteration and it checks the $Error_{rad-lw}$. If it is not under TOL_{rad-lw} then it is set to propagation mode $PMODE = OFF$, so that only the already called WALLs are used in the next iteration. In doing so, it first tries a local iteration. The new test values are the $q_{rad-lw,j[side]}^{new}$ for those rooms where all their walls have been called (ZONE 27), and for those neighbouring rooms in the propagation (18, 24 and 26) the new test values are obtained (Appendix 2).

Finally, the RAD2WALLs elements are in charge of calling the individual WALLs called in a room (ZONES 18, 24 and 26) or all the WALLs of a room (ZONE 27). RAD2WALLs emit signals which plug into the q_{rad-lw} port of WALLs.

Now, WALLs clear for the new surface temperatures which force the heat power balance at both sides of each WALL without state transition since we are still trying to solve the effects of the event.

- step (2) The local iteration continues over the subset of all the $q_{rad-lw,j[side]}$ which belong to WALLs already called. However, since we are trying to solve a non-linear equation system where some variables are kept fixed, it is possible that the $Error_{rad-lw}$ cannot be reduced below TOL_{rad-lw} . This is checked by GLOBALBALANCE using the condition of a non-monotonically decreasing of $Error_{rad-lw}$ after each iteration.
- step (3) If this happens, then GLOBALBALANCE sets propagation mode $PMODE = ON$ and all the WALLs of the neighbouring rooms are called. This means computing the successive state transition of new WALLs and the appearance of a new 'propagation front'.
- step (4) If during a propagation it happens that $Error_{rad-lw} < TOL_{rad-lw}$ then the iteration is finished and the WALLs are called with the final $q_{rad-lw,j[side]}^{new}$.

Finally each WALL stores its new converged state and sends a signal to their ZONES indicating the new convective heat power. The QSS integrator schedules its new time advance for the next temperature event from the ZONE.

It is worthwhile to make a few remarks. Firstly, iterating over all the WALLs always gives a solution since both the number of equations and unknowns match, but it would make the computation very time consuming. Some parts of the building, away from the event, may be changing slowly. If these elements are included in the iteration from the beginning then one may check too late that calling them was actually not necessary. This would be a way to deal with the thermal coupling sub-problem. Secondly, when iterating locally, if condition (11) is attained the

solution is not considered good whenever the GLOBALBALANCE is not in propagating mode. That means we need to check if the elapsed time has changed enough the state of the WALLs at the 'propagation front' so that the apparently good solution for the $q_{rad-lw,j[side]}$, in fact, does not yet satisfy the superficial heat power balance when this time lapse is taken into account. This would deal with the relaxation sub-problem. Notice that in fact, there is a trade-off in a building with free floating temperatures. If the elapsed time between events is short, the time effect is less important for a 'propagation front' and less rooms are called but at the cost of a higher number of events. On the contrary, if the elapsed time is big, the thermal relaxation of the walls may be important and the front propagates further away from the element where the event was originated but less events are emitted. Nevertheless, in general, when a WALL is near a steady state condition (i.e. not strong dynamics) and an 'iteration propagation front' arrives at it, the WALL actually may stop the need for further propagation.

In the next section, we try to illustrate all the findings and comments done in this section by means of some simulation examples.

4. Example and discussion

All the example buildings have the constructive solutions shown in Table 2.

The vertical partition wall, roof/floor and external wall dynamics were computed with; 1089, 3801 and 2392 poles respectively, which are negative numbers distributed unevenly depending on the wall composition. The k -pole absolute value is α_k ($\alpha_k < \alpha_{k+1}$) and the actual amount of poles used for the successive transition state calculation was always $k = 400$ (i.e. $k \leq 400$, the rest $400 < k$ was employed to compute the terms Q^m (see Appendix 1 and Francés, Escrivá, and Ojer 2014 for details)). We must stress that not all these 400 states (at both sides of each wall) are permanently tracked during the simulation (see Francés, Escrivá, and Ojer 2014 for details). This value represents the maximum amount of retained data used to characterize a certain wall-type dynamics. In other words, this is the information at our disposal, in order to be able to construct a heat flux response after a superficial temperature change event. However, not all

Table 2. Constructive solutions. The order is from outside to inside (the floor is the same as the roof but the order of the layers is reversed).

Name	k [W/m/K]	ρ [kg/m ³]	c_p [J/kg/K]	Thickness[m]
<i>External Wall</i>				
Solid brick	0.870	1800.0	1380.0	0.120
Concrete mortar	1.400	2000.0	1050.0	0.015
Polystyrene foam	0.033	25.000	837.00	0.040
Void brick	0.490	1200.0	920.00	0.040
Gypsum	0.300	800.00	920.00	0.015
<i>Roof/Floor</i>				
Pavement	1.100	2000.0	1380.0	0.050
Waterproof film	0.190	1100.0	1680.0	0.010
Light concrete	0.350	1000.0	1050.0	0.100
Concrete filler block	1.540	1254.0	1050.0	0.260
Gypsum	3.000	800.00	920.00	0.020
<i>Vertical partition wall</i>				
Gypsum plaster	0.400	1000.0	1000.0	0.010
Void brick	0.210	630.0	1000.0	0.070
Gypsum plaster	0.400	1000.0	1000.0	0.010

the information must be used at each step. Roughly, by looking at Appendix 1, Equations (A6b), it can be concluded that if the elapsed time t_e is large enough then many response modes are negligible (those with big enough α_k). These response modes contribute practically nothing to the overall response and can be discarded beforehand. In this way, the effort to compute the response becomes adaptive. Stated in another way, if the next superficial temperature change occurs close in time then the computational effort increases and vice versa. Therefore an obvious question comes up, what is the shorter t_e ? (see details in Francés, Escrivá, and Ojer 2014). The minimum elapsed time t_e for each constructive solution that allows to model correctly the conduction heat transfer dynamics was 0.26[s], 9.7[s] and 2.8[s] for the vertical partition wall, the roof/floor and the external wall respectively.

The DRYCOLD weather file, of the ASHRAE (2001) procedure for energy simulation tools, was used. It has a very cold winter and hot summer and therefore represents a big forcing or excitation function for any building. Our tests were done assuming that the temperature of the building was floating, that is, there was no thermostat. The shortwave length emissivity at both sides is 0.85 and the infra-red (longwave length) has been kept at 0.8 inside and 0.9 outside for all the WALL elements. The internal convection model used was the same as the simple model of EnergyPlus (2012) (inside: $3.076 [\text{Wm}^{-2}\text{K}^{-1}]$ for walls, $0.948 [\text{Wm}^{-2}\text{K}^{-1}]$ blocked natural convection, $4.04 [\text{Wm}^{-2}\text{K}^{-1}]$ non-blocked for ceiling and floor, outside: $10.79 [\text{Wm}^{-2}\text{K}^{-1}]$). No optimization of the code was done in terms of computer speed. Finally, the tolerance for the infra-red heat exchange was $TOL_{rad-lw} = 0.01 [\text{W}]$.

The main parameter that controls the simulation is the quantum for the energy (temperature) change in the zone, obviously along with the type of QSS method. We have chosen the same quantum for all the zones. The aim of the paper is not to make a complete study of the QSS methods but to explore the potentialities, difficulties and weaknesses, however, some extra comments are needed. Bolduc and Vangheluwe (2003) made a discussion about the consistency, convergence and stability of quantization methods and proposed an algorithm to make the quantum adaptive. The main problem comes when $\dot{x}_n = 0$ for any n of the ODEs. Non-autonomous systems are problematic since whenever the derivatives are zero, convergence is undefined (see Bolduc and Vangheluwe 2003). Notice that by assuming that time $z = t$ is another independent state and adding an equation like $\dot{z} = 1$ to the differential equations system, we make the system autonomous but now the existence of a quantum of time makes the method resemble a discrete-time method.

The QSS methods employed here are those implemented into the software PowerDEVS (Bergero and Kofman 2011). The details and differences among the basic QSS method and its variants CQSS or LIQSS can be found elsewhere (see Migoni 2010, PhD thesis). Implemented QSS methods could be considered, in general, as not adaptive. They use an absolute and a relative quantum (Kofman 2007). It should be pointed out that these quantized integrators do not consider time as another state in the case of non-autonomous systems. Here, we have not used the relative quantum but just a fixed one for the QSS and CQSS integrators. CQSS is used when there is a need to correctly represent oscillatory dynamics (see Migoni 2010). However, LIQSS (a

linearly implicit QSS method) goes a step further from QSS and CQSS and tries to find when the time derivative of a state is going to be zero and changes the quantum size accordingly in order to reach that equilibrium state. In case of a non-linear system LIQSS only gets an approximation to that point. Therefore, LIQSS might be considered as a kind of adaptive method. The original deduction of LIQSS (Migoni 2010) was done for autonomous systems. In Appendix 3 we have adapted LIQSS for non-autonomous systems. LIQSS's objective is to avoid high frequency oscillations when any of f_j at the right hand side of the system in Equation (6) is close to zero.

In Figure 6 we show the temperature for 5 of the 27 ZONES in a $3 \times 3 \times 3$ -building. It was solved using a LIQSS integrator with a quantum of 3000 [J] for the energy content of the air which represents air temperature changes of around $0.1[^\circ\text{C}]^2$. The zone numbers correspond to those of Figure 5. As expected, the core zone (number 14) has a smoother temperature fluctuation than the others since it is protected from the exterior. It is followed by zones 1 and 2 which are at the 1st floor and finally zones 21 and 23 -in this order- which are at the 3rd floor and are more exposed. In Figure 6(D) a day is plotted with bars. It is clearly appreciated that the sampling rate is not uniform like in traditional methods but adjusts itself to the speed of the temperature changes. Near the top and bottom of the temperature profiles or in the plateaux, when the convection power is nearly zero, there appears some small ripples (see the detail in sub-figure D). We think that this is due to LIQSS failing to find the exact zero and to the non-autonomous character of the problem. When the quantum is reduced these ripples are smoothed out since the time dependency is reduced. This is a general trend for any QSS method. Figure 7 shows the effect of decreasing the quantum on the ripples or oscillations.

Table 3 collects information about several yearly simulations. Notice the high computing time values. In general, the smaller the quantum the greater the precision but the greater the amount of events and computational time. It can be seen that LIQSS seems to be superior to QSS or CQSS since for the same quantum (roughly the same precision) it takes less time. This happens despite LIQSS involves more calculations. Table 3 also shows how the quantum and the QSS method affect the mean value of the number of rooms being called per concurrent iteration. For the $3 \times 3 \times 3$ and $1 \times 27 \times 1$ cases, the maximum is 27 rooms while for the $5 \times 1 \times 6$ -building the maximum is 30 rooms. As a general trend, the greater the quantum the closer the mean is to the maximum. The cause is clear, a big quantum lets higher time advances and therefore the dynamical relaxation of the conduction elements is not negligible. As a check of this statement, Table 3 collects two cases where the core room of the $3 \times 3 \times 3$ -building has been kept at a fixed and low temperature ($2[^\circ\text{C}]$) in order to simulate the effect of several conduction elements close to a steady state. It is shown how the average number of called rooms decreases and that the effect is bigger for a smaller quantum.

Figure 8 represents the detailed histograms about the number of called rooms for a state transition per concurrent iteration. In general, almost all the rooms are called. However, it must be taken into account that the buildings are relatively small, the temperature of all the rooms is floating -there is no active thermostat- and the BESTEST weather is an extreme case. The

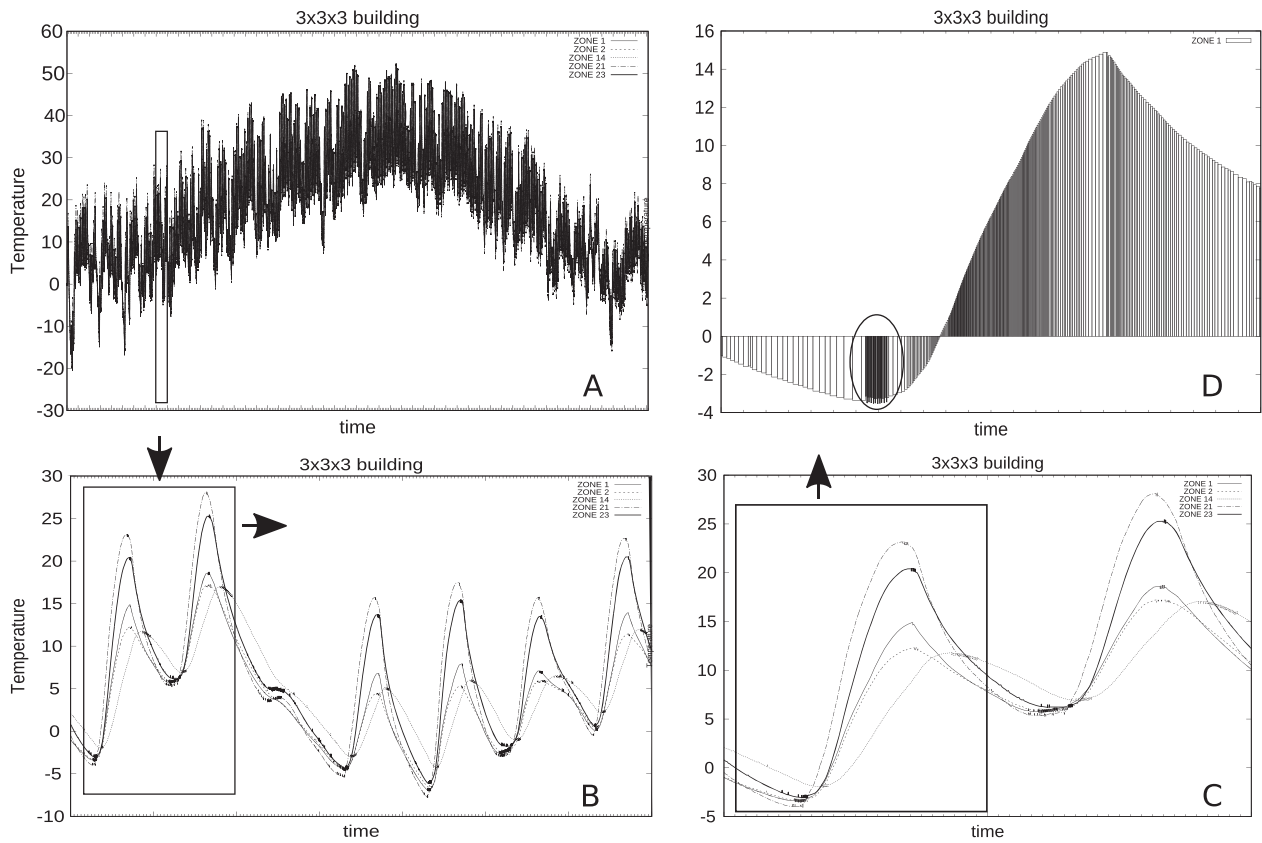


Figure 6. 3 × 3 × 3-Building. LIQSS DEVS integrator quantum = 3000[J]. Zone temperature evolution. (A) year (B) a week (C) two days (D) one day.

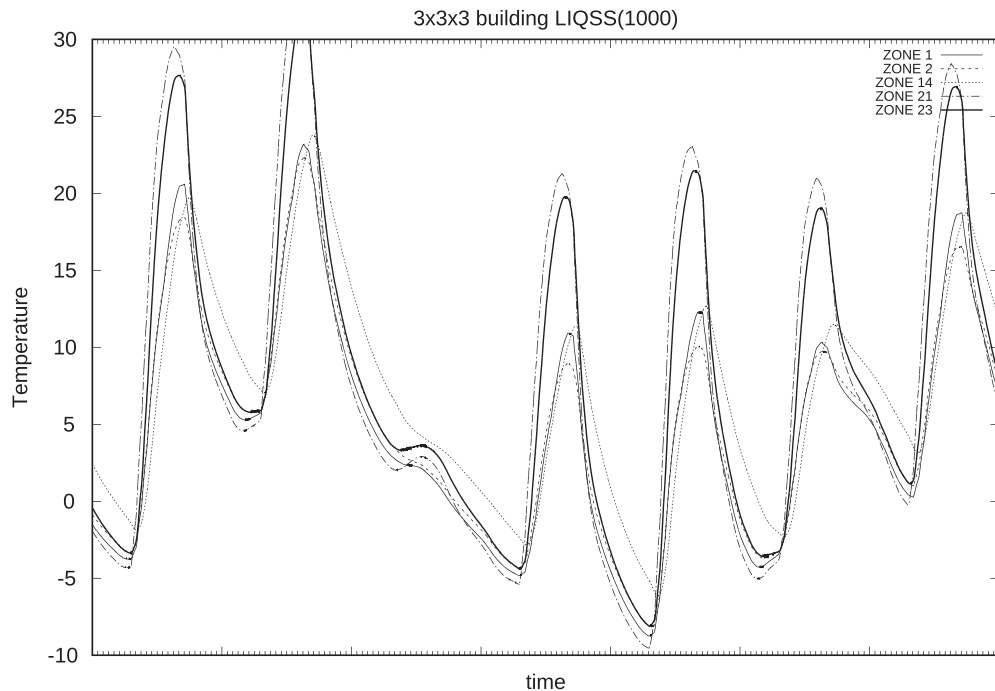


Figure 7. 3 × 3 × 3-Building. Same week as (B) of Figure 6 but with a smaller quantum 1000[J].

design of our test building was aimed to check if the internal zone was called much less than the external ones.

Some interesting trends show up. The case with the smallest number of called rooms per iteration is LIQSS with a quantum equal to 1000[J] (with or without the core room temperature

fixed). Besides the obvious case of calling all the rooms, there are others where it is noticeable that a smaller number than the maximum number of rooms was called. The reason is also clear and is related to where the event originates. Next there is a list of the number of rooms called per building case:

Table 3. Summary of outcomes from different building simulation cases using different QSS methods and quantum values.

QSS method	quantum [J]	mean size	time [s]	relative time	Building case
LIQSS	3000	25.37	5210	1.00	3x3x3
LIQSS*	3000	25.35	6010	1.15	–
LIQSS	1000	21.19	17500	3.36	–
LIQSS*	1000	20.60	16800	3.22	–
QSS	33000	26.93	8850	1.00	–
QSS	11000	26.87	11800	1.33	–
QSS	3000	26.32	28800	3.25	–
CQSS	33000	26.86	13200	1.00	–
CQSS	11000	26.77	17100	1.30	–
CQSS	3000	26.23	30000	2.27	–
LIQSS	3000	24.88	7600	1.00	1x27x1
LIQSS	3000	27.08	6990	1.00	5x1x6
LIQSS	1000	24.36	18400	2.63	–

Note:(*) constant temperature of the core zone (number 14). (note: CPU intel i7.6500U, SSHD).

- $3 \times 3 \times 3$ case (see Figure 5):
 - 4 : corresponds to an event at a room at the corner of the building (i.e ZONES 1,3,7,9,19,21,25,27).
 - 5 : corresponds to an event at a room at the edge of the building (i.e. ZONES 2,4,6,8,10,12,18,16,20,22,24,26)
 - 6 : corresponds to an event at a room at the centre of the face of the building (i.e. ZONES 11,13,17 and 15).
- $1 \times 27 \times 1$ case:
 - 3 : are the rooms in the middle of this long building (25 rooms).
- $5 \times 1 \times 6$ case:
 - 3 : rooms when the event originates at the corners (4 rooms).
 - 4 : rooms when the event originates at the edge (14 rooms).
 - 5 : rooms when the event originates at the centre (12 room).

The relative distribution in Figure 8 agrees with the relative amount of potentially called rooms at each position.

4.1. Main outcome

Although we think that the paper introduces several interesting and positive ideas to achieve a high performance event-driven simulation, it is not a successful implementation. As Figure 8 shows, some degree of *asynchronicity* is actually obtained but the computing times are very high (see Table 3). This would make, at first sight, the method inoperative.

We have identified the main reason. Obviously, it seems that every event triggers the state transition of too many wall elements. Moreover, the short elapsed time (or high time frequency, in the order of few minutes or even seconds) causes a computational penalty on the SST, undermining its high potential to speed up the computations since the SST method performs much better for long elapsed time between calls.

But, what are the causes of calling too many wall elements so frequently?

The difficulties lie on using the traditional or conventional heat transfer functions for the wall elements. The surface temperature dynamics (time derivative) of the walls is not present *explicitly* in the discrete-event model. This transforms the original system of differential equations into a non-autonomous one. Moreover, it forces to devise a concurrent iteration to solve a non-linear algebraic system of equations for several rooms. Figure 4 shows that only the dynamics of the thermal zones/rooms is explicitly included (represented by the air state, -energy or temperature-). The conventional transfer functions take the surface temperatures as the internal wall state variables. This makes the DEVS integrator (QSS) unaware of their change. However, the QSS does need *explicitly* their rate of change or, so to speak, to appear as external state variables (i.e. *on the QSS*

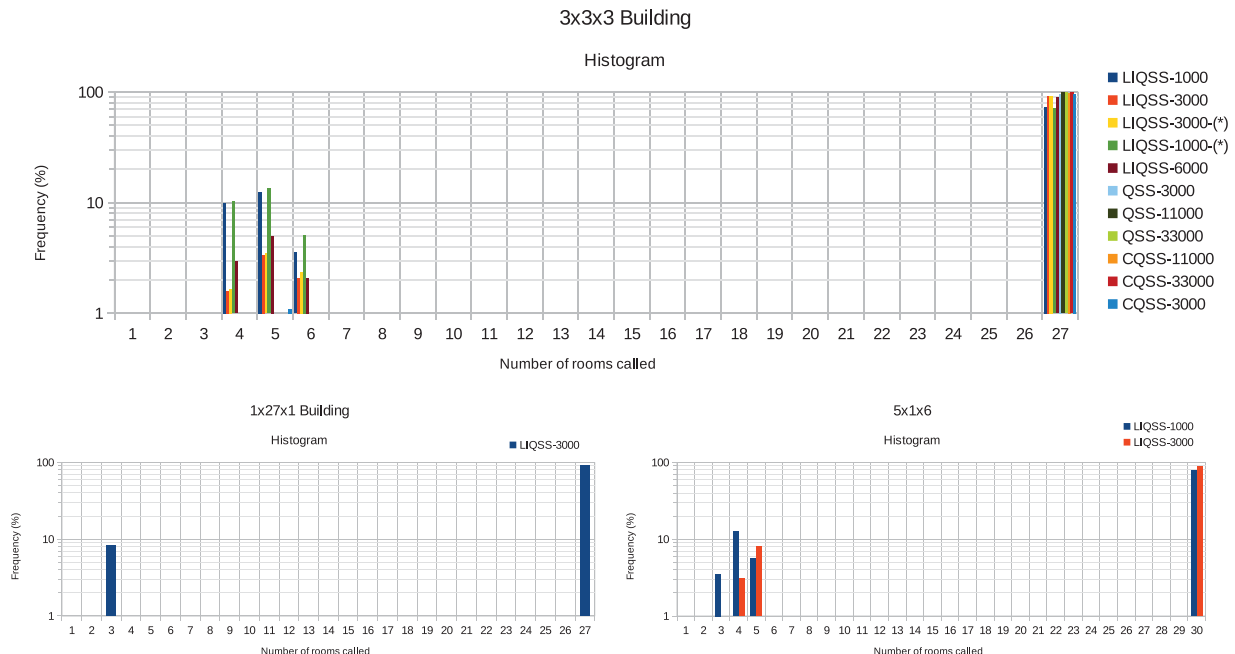


Figure 8. Histogram or ratio of number of iterations finished with a certain number of rooms called during a concurrent iteration to the total number of iterations during a yearly simulation.

side). The derivatives $\{\dot{T}_0, \dot{T}_1\}$ should be handed over to the discrete-event model integrator.

Therefore a new point of view is required. We propose abandoning the traditional transfer functions (see Equation (A2)) and develop a new scheme that uses the following \hat{G} instead, along with the SST method and a DEVS model using QSS:

$$\begin{bmatrix} \mathcal{L}\dot{T}_0(s) \\ \mathcal{L}\dot{T}_1(s) \end{bmatrix} = \begin{bmatrix} \hat{G}_{00}(s) & \hat{G}_{01}(s) \\ \hat{G}_{10}(s) & \hat{G}_{11}(s) \end{bmatrix} \begin{bmatrix} \mathcal{L}q_0(s) \\ \mathcal{L}q_1(s) \end{bmatrix} \quad (12)$$

Now input and output signals are different. The input signals would be the conduction heat fluxes while the output signals *would not be* the surface temperatures but their time derivatives. This decision would allow including the ‘dynamics’ of the surface temperatures, of the multi-layered slabs, as additional new \dot{x} variables into the scheme of Figure 4. Their role would be on an equal footing to that of the zone states. Besides, with this modification, as the surface temperatures would be outputs from the QSS integrators, there would not exist a need to concurrently solve a non-linear algebraic system of equations for several zones. The conduction heat fluxes would be linearly interpolated between events and the successive state transition would compute the state transition of the temperature time derivative due to the change of these fluxes. In short, new conduction heat transfer functions must be employed as a means to get the time derivatives of the surface temperatures of the walls.

A final remark is worthwhile. The reason to initially pose the formulation in the way described by the paper was to *reduce* the amount of state variables. Let us take the $3 \times 3 \times 3$ example building: the number of its state variables, according to our formulation, is 27 (the room temperatures). If the surface temperatures were also included as state variables, according to the new proposal, then that amount would be 243.

The paper shows that although asynchronicity can be obtained by our formulation, the potential advantage of a *reduced* problem is not achieved. As it was shown, the root cause is that our formulation becomes non-autonomous and sensitive to the time evolution of the surface temperatures. By taking a careful look at our results we noticed that the problem was a bit more convoluted. These temperatures are forced to vary linearly (between events) like traditional heat transfer functions do. However, their actual time evolution is exponential-like. In Ojer et al. (2015), as aforementioned in the introduction, we showed that this linearity comes along with a hidden effect. It gives an extra fictitious heat capacity to the wall element, due to the mismatch among the surface heat fluxes between sampling time-points. Fortunately, this mismatch cancels out and simulations do not crash³. Logically, the greater the elapsed time, the greater the effect. On the other side, searching for higher computational speeds implies increasing the quantum size since the elapsed time between events gets bigger. Nevertheless, the latter promotes the previous spurious effect. As a consequence, there appears a superimposed oscillation of the room temperatures (like a predictor-corrector step) which unfortunately increases the number of events and decreases the overall performance and accuracy. As a consequence, this compelled us to use small quantum sizes thus increasing the computational cost.

5. Conclusions

This article has explored a discrete-event model for yearly energy simulation of buildings. Traditional methods are time-driven simulations with a fixed time step. Our strategy to shift the current paradigm was to proceed in a step-by-step manner by reusing well-based traditional knowledge as much as possible. Therefore the paper uses the well-known conduction heat transfer functions which relate wall superficial temperatures to conduction heat fluxes at both sides.

The traditional methods, employing these functions, do not allow a variable time step needed by a discrete-event simulation. Therefore the paper employs a less conventional method based on the Japanese school. It is a Direct Root Finding based method, named successive state transition method (SST) which does allow variable time steps. A previous work for a single zone (see Francés, Escrivá, and Ojer 2014), pointed out that the SST and QSS methods work very well together. Thus, our intention here was to discover if it could be extended straight away to multi-zone buildings. The answer is negative.

On one side, the paper confirms that event-driven simulations within the building energy field, are possible along with Direct Root Finding and different QSS methods. Incidentally, the paper also proposes an improvement for the LIQSS method applied to non-autonomous systems. Moreover, we emphasize the positive features of the SST method. It allows the excitation of a variable amount of internal states, as a function of the elapsed time between calls. Therefore, its workload is adaptive: the longer the elapsed time, the smaller the computational effort. In other words, if the walls were called at a low frequency, then the computational burden would decrease without affecting the accuracy. Nevertheless, despite the model was implemented without optimizations (like file writing or detailed analysis of execution time), the expected computational load reduction was not achieved.

However, this downside has a positive facet since led us to another important finding: the traditional conduction heat transfer functions do not actually fit into discrete-event simulations. The new paradigm requires to shift the multi-layered wall heat transfer functions from Equations (A2) to the new set (12). The new functions relate the input heat fluxes to the time derivative of the superficial temperatures. This new view is not so obvious as might seem at first sight and implies also a change in the traditional way of thinking.

The development and implementation details of this new paradigm remains open for further research. In doing so, the computation could benefit from specialized QSS solvers for ODEs (see Fernandez and Kofman 2014) and even parallelizing techniques (see Fernandez, Kofman, and Bergero 2017).

Notes

1. Precisely, this happened to be the main obstacle for the success of our implementation, as it will be shown.
2. The room mass of dry air is $m_{da} = 45[\text{m}^3] \times 1.2[\text{kg} \cdot \text{m}^{-3}] = 54[\text{kg}]$ and a quantum in energy is related to a quantum in temperature as $\Delta Q_E = m_{da} \cdot c_{da} \cdot \Delta Q_T$, then $\Delta T \approx 0.1[^\circ\text{C}] = \Delta Q_T \cdot 2$ since the quantum ΔQ represents one half of the magnitude change.
3. Notice that this also happens in accepted traditional simulation methods based on the same assumptions.
4. (0 \equiv origin of spatial axis and 1 \equiv the other end side).

Acknowledgements

We would like to acknowledge the help in using the software PowerDevs (Bergero and Kofman 2011) to Federico Bergero.

Disclosure statement

No potential conflict of interest was reported by the author(s).

ORCID

Víctor-Manuel Soto-Francés  <http://orcid.org/0000-0002-0244-2668>

Emilio-José Sarabia-Escrivá  <http://orcid.org/0000-0001-9873-3138>

José-Manuel Pinazo-Ojer  <http://orcid.org/0000-0001-8835-3318>

Pedro-Juan Martínez-Beltrán  <http://orcid.org/0000-0002-6734-5500>

References

- ANSI/ASHRAE Standard 140-2001. 2001. "Standard Method of Test for the Evaluation of Building Energy Analysis Computer Programs." ASHRAE Inc., 1791 Tullie Circle NE Atlanta, GA 30329. www.ashrae.org. <https://webstore.ansi.org/Standards/ASHRAE/ANSIASHRAE1402001>
- Bergero, F. M., F. Casella, E. Kofman, and J. Fernández. 2017. "On the Efficiency of Quantization-based Integration Methods for Building Simulation." *Building Simulation* 11 (2): 405–418. doi:10.1007/s12273-017-0400-1.
- Bergero, F. M., and E. Kofman. 2011. "Powerdevs. a Tool for Hybrid System Modeling and Real Time Simulation." *Simulation: Transactions of the Society for Modeling and Simulation International* 87 (1–2): 113–132.
- Bolduc, J. S., and H. Vangheluwe. 2003. "Mapping ODEs to DEVS; Adaptive Quantization." In A. Bruzzone & Mhamed Itmi editors. *Summer Computer Simulation Conference, Montreal, Canada*, July, 401–407. Society for Computer Simulation International (SCS).
- Chen, Y., and S. Wang. 2001. "Frequency-domain Regression Method for Estimating Ctf Models of Building Multilayer Constructions." *Applied Mathematical Modelling* 25 (7): 579–592. doi:10.1016/S0307-904X(00)00067-6.
- EnergyPlus Engineering Reference. 2012. "U. S. D. of Energy." <http://apps1.eere.energy.gov/buildings/energyplus/>, <http://apps1.eere.energy.gov/buildings/energyplus/>.
- Fernandez, J., and E. Kofman. 2014. "A Stand-alone Quantized State System Solver for Continuous System Simulation." *Simulation* 90 (7): 782–799. doi:10.1177/0037549714536255.
- Fernandez, J., E. Kofman, and F. Bergero. 2017. "A Parallel Quantized State System Solver for ODEs." *Journal of Parallel and Distributed Computing* 106: 14–30. doi:10.1016/j.jpdc.2017.02.011.
- Francés, V. M. S., E. S. Escrivá, and J. M. P. Ojer. 2014. "Discrete Event Heat Transfer Simulation of a Room." *International Journal of Thermal Sciences* 75: 105–115. doi:10.1016/j.ijthermalsci.2013.07.024.
- Francés, V. M. S., E. J. S. Escrivá, and J. M. P. Ojer. 2015. "Discrete Event Heat Transfer Simulation of a Room Using a Quantized State System of Order Two, QSS2 Integrator." *International Journal of Thermal Sciences* 97: 82–93. doi:10.1016/j.ijthermalsci.2015.06.006.
- Francés, V. M. S., E. S. Escrivá, and J. M. P. Ojer. 2016. "A Hygrothermal Dynamic Zone Model for Building Energy Simulation." *Energy and Buildings* 133: 389–402. doi:10.1016/j.enbuild.2016.10.002.
- Functional Mock-up Interface Standard. <http://fmi-standard.org/>.
- Goldstein, R., S. Breslav, and A. Khan. 2013. "Using General Modeling Conventions for the Shared Development of Building Performance Simulation Software." In *Proceedings of BS2013: 13th Conference of International Building Performance Simulation Association*, Chambéry, France, August 26–28.
- Gunay, H. B., W. O'Brien, I. Beausoleil-Morrison, R. Goldstein, S. Breslav, and A. Khan. 2014. "Coupling Stochastic Occupant Models to Building Performance Simulation Using the Discrete Event System Specification Formalism." *Journal of Building Performance Simulation* 7 (6): 457–478. doi:10.1080/19401493.2013.866695.
- Gunay, H. B., L. O'Brien, R. Goldstein, S. Breslav, and A. Khan. 2013. "Development of Discrete Event System Specification (DEVS) Building Performance Models for Building Energy Design." In *Proceedings of the Symposium on Simulation for Architecture & Urban Design SimAUD '13*. San Diego, California. ISBN 978-1-62748-035-2.
- Hayashi, T., Y. Urano, T. Watanabe, and Y. Ryu. 1985. "Passive System Simulation Program PSSP and its Applications." In *IBPSA (Ed.), Building Simulation, International Building Performance Simulation Association, 1985*, 346. <http://www.ibpsa.org>.
- Hittle, D. C., and R. Bishop. 1983. "An Improved Root-finding Procedure for Use in Calculating Transient Heat Flow Through Multi-layered Slabs." *International Journal of Heat and Mass Transfer* 26 (11): 1685–1693. doi:10.1016/S0017-9310(83)80089-1.
- IEA-annex60. "New Generation Computational Tools for Building and Community Energy Systems based on the Modelica and Functional Mockup Interface Standards." <http://iea-annex60.org/>.
- Ko, Y., and S. T. No. 2015. "A Study on Comparison of Building Energy Simulation and Measurement Results for a City Hall." *Journal of Building Construction and Planning Research* 3: 1–9. doi:10.4236/jbcpr.2015.31001, <http://www.scirp.org/journal/jbcpr>.
- Kofman, E. 2007. "Relative Error Control in Quantization based Integration." In *XII Reunión de Trabajo en Procesamiento de la Información y Control*, 16–18 Octubre. <http://www.fceia.unr.edu.ar/kofman/files/lisd0703.pdf>.
- Li, N., Z. Yang, B. Becerik-Gerber, C. Tang, and N. Chen. 2015. "Why Is the Reliability of Building Simulation Limited As a Tool for Evaluating Energy Conservation Measures?" *Applied Energy* 159: 196–205. doi:10.1016/j.apenergy.2015.09.001.
- Maestre, I. R., P. R. Cubillas, and L. Pérez-Lombard. 2010. "Transient Heat Conduction in Multi-layer Walls: An Efficient Strategy for Laplace's Method." *Energy and Buildings* 42: 541–546.
- Mazzarella, L., and M. Pasini. 2015. "Conduction Transfer Function vs Finite Difference: Comparison in Terms of Accuracy, Stability and Computational Time." In *6th International Building Physics Conference, Energy Procedia Ed. Elsevier, IBPC (2015)*. http://www.ibpc2015.org/app/media/upload_s/files/papers/IBPC15_ID647_FinalX.pdf.
- Migoni, G. 2010. "Simulacion por Cuantificacion de Sistemas Stiff." PhD thesis, Facultad de Ciencias Exactas, Ingenieria y Agrimensura. Universidad Nacional de Rosario.
- Ojer, J. M. P., V. M. S. Francés, E. S. Escrivá, and L. S. Francés. 2015. "Thermal Response Factors to a 2nd Order Shaping Function for the Calculation of the 1D Heat Conduction in a Multi-layered Slab." *International Journal of Heat and Mass Transfer* 88: 579–590. doi:10.1016/j.ijheatmasstransfer.2015.04.110.
- Pakanen, J. 1996. "Conduction of Heat Through One- and Multilayer Slabs: A Differential-difference Approach." *Numerical Heat Transfer, Part B: Fundamentals* 30:
- Spawn of EnergyPlus. "Design and Implementation (Working Draft)." <https://bl-srg.github.io/soep/>.
- Trcka, M., J. L. M. Hensenaand, and M. Wetter. 2009. "Co-simulation of Innovative Integrated HVAC Systems in Buildings." *Journal of Building Performance Simulation* 2 (3): 209–230.
- Wainer, G. A., and P. J. Mosterman. 2010. *Discrete-Event Modeling and Simulation: Theory and Applications*. Boca Raton: CRC Press. ISBN 9781420072334.
- Wang, H., and Z. J. Zhai. 2016. "Advances in Building Simulation and Computational Techniques: A Review Between 1987 and 2014." *Energy and Buildings* 128: 319–335. doi:10.1016/j.enbuild.2016.06.080, <http://www.sciencedirect.com/science/article/pii/S0378778816305692>.
- Wetter, M., and T. S. Nouidui. 2015. "Overview of SOEP." In *Laurence Berkeley National Laboratory (Simulation Research Group)*. <http://simulationresearch.lbl.gov/modelica/downloads/workshops/2015-06-22-lbnl/slides/soep-overview.pdf>.
- Yoshimi, U., and W. Toshiyuki. 1981. "An Analysis of Multi-layer Wall Heat Transfer by State Transition Matrix: Part 1 An Approximate Transfer Functions Model and Its Accuracy." *Transactions of the Architectural Institute of Japan* 305: 97–111. <http://ci.nii.ac.jp/naid/110003881844/en/>.
- Yoshimi, U., and W. Toshiyuki. 1982. "An Analysis of Multi-layer Wall Heat Transfer by State Transition Matrix: Part 2 a Successive Calculation Method and Its Accuracy." *Transactions of the Architectural Institute of Japan* 311: 57–66. <http://ci.nii.ac.jp/naid/110003881951/en/>.
- Ziegler, B. P., Herbert Praehofer, and Tag Kim. 2000. *Theory of Modeling and Simulation*. 2nd ed. San Diego, California, USA: Academic Press. ISBN:978-0127784557.
- Zimmermann, Gerhard 2001. "A New Approach to Building Simulation Based on Communicating Objects." In *Seventh International IBPSA Conference Rio de Janeiro, Brazil* August 13–15, 2001.

Appendices

Appendix 1. Successive transition state

The successive transition method can be tracked back to Urano et al. (see Yoshimi and Toshiyuki 1981, Yoshimi and Toshiyuki 1982) or Testuo Hayashi et al. (see Hayashi et al. 1985). A short resume of the method is the following. The response in heat flux on a surface of a multi-layered wall to a unitary (step) temperature excitation is expressed as:

$$\phi^m(t) = A_0^m + \sum_{k=1}^{\infty} A_k^m e^{-\alpha_k t}, \quad (A1)$$

where $m \in \{X, \pm Y, Z\}$. The well-known matrix Chen and Wang (2001) expression for the heat fluxes (W/m^2) in Laplace domain, in our case is⁴:

$$\begin{bmatrix} \mathcal{L}q_0(s) \\ \mathcal{L}q_1(s) \end{bmatrix} = \begin{bmatrix} \mathcal{G}_X(s) & \mathcal{G}_{-Y}(s) \\ \mathcal{G}_Y(s) & \mathcal{G}_Z(s) \end{bmatrix} \begin{bmatrix} \mathcal{L}T_0(s) \\ \mathcal{L}T_1(s) \end{bmatrix}. \quad (A2)$$

The A_0^m, A_1^m, \dots in Equation (A1), are the residuals at the corresponding poles $-\alpha_k$. These poles have been calculated using the Hittle method (see Hittle and Bishop 1983) and characterize completely the dynamic thermal response of the element. The greater the k the biggest the α_k becomes. There are four A sets, well in fact three since $A^{-Y} = -A^Y$. In all of them A_0 is the thermal conductance of the element. The infinite series cannot be computed so the series must be truncated at $k = K$.

$$\phi^m(t) = A_0^m + \sum_{k=1}^K A_k^m e^{-\alpha_k t} + Q^m \delta(t) \quad (A3)$$

In Equation (A3) δ is the Dirac's function and $Q^m = \sum_{k=K+1}^{\infty} A_k^m / \alpha_k$. Finally the transfer function $\mathcal{G}_m(s)$ is obtained multiplying the Laplace transformation of $\phi^m(t)$ by s .

$$\mathcal{G}_m(s) = A_0^m + \sum_{k=1}^{K_0} \frac{A_k^m s}{s + \alpha_k} + Q^m s \quad (A4)$$

The hold or shaping function between sampled values of the forcing temperatures is not a triangle but a trapezoidal shape. This shape allows to adapt itself to a variable time step. Multiplying Equation (A4) by the transfer function of the hold function, using the Z-transform and after some involved manipulations, we arrive at the different equations about the heat fluxes at each side of the multi-layered element:

$$\begin{aligned} q_0(t_n) &= a_{00}T_0(t_n) + a_{01}T_1(t_n) + D_0 \\ q_1(t_n) &= a_{10}T_0(t_n) + a_{11}T_1(t_n) + D_1 \end{aligned} \quad (A5)$$

The Equation (A5) is just a linear system of equations relating the new temperatures and the density of the conduction heat flux [W/m^2] at both sides and at t_n . The coefficients a_{xx} and D_x depend on the time elapsed t_e from the last evolution of the states and their previous value $w_{k,n-1}^{side}$, $side \in \{0, 1\}$, $k \in \{1, \dots, K\}$. The successive state transition computation has been re-organized as follows to adapt it to the DEVS model implementation:

$$\varphi_k = e^{-\alpha_k t_e} \quad (A6a)$$

$$p_k = (1 - \varphi_k) / \alpha_k t_e \quad (A6b)$$

$$a_{00} = A_0^X + \frac{Q^X}{t_e} + pA^X \quad (A7a)$$

$$a_{01} = -A_0^Y - \frac{Q^Y}{t_e} - pA^Y \quad (A7b)$$

$$a_{10} = -a_{01} \quad (A7c)$$

$$a_{11} = A_0^Z + \frac{Q^Z}{t_e} + pA^Z \quad (A7d)$$

$$b_{00} = -\left(pA^X + \frac{Q^X}{t_e}\right) \quad (A7e)$$

$$b_{01} = \left(pA^Y + \frac{Q^Y}{t_e}\right) \quad (A7f)$$

$$b_{10} = -b_{01} \quad (A7g)$$

$$b_{11} = -\left(pA^Z + \frac{Q^Z}{t_e}\right) \quad (A7h)$$

$$D_0 = b_{00}T_{0,n-1} + b_{01}T_{1,n-1} + W_{\varphi}^0 \quad (A7i)$$

$$D_1 = b_{10}T_{0,n-1} + b_{11}T_{1,n-1} + W_{\varphi}^1 \quad (A7j)$$

with $pA^m = \sum_{k=1}^K p_k A_k^m$ and $W_{\varphi}^x = \sum_{k=1}^K \varphi_k W_{k,n-1}^x$, $side \equiv x \in \{0, 1\}$. Recall that not all the terms in Equation (A6b) are computed. It depends on how much time has elapsed t_e and the value of the poles since φ_k goes to zero as time goes by (For details see Francés, Escrivá, and Ojer 2014).

Equations (A8) serve to update the states at both sides of the conduction element.

$$w_{k,n}^0 = \varphi_k W_{k,n-1}^0 + p_k \{(A_k^X T_{0,n} - A_k^Y T_{1,n}) - (A_k^X T_{0,n-1} - A_k^Y T_{1,n-1})\} \quad (A8a)$$

$$w_{k,n}^1 = \varphi_k W_{k,n-1}^1 + p_k \{(A_k^Y T_{0,n} + A_k^Z T_{1,n}) - (A_k^Y T_{0,n-1} + A_k^Z T_{1,n-1})\} \quad (A8b)$$

Appendix 2. Projection method for $q_{rad-lw,new}$ test values

Any set $\{q_{rad-lw,j[side]}\}$ for a n -room is constrained to fulfil Equation (A9) or, in other words, the radiation heat power balance.

$$\sum_{j \in B_n} q_{rad-lw,j[side]} = 0 \quad (A9)$$

When propagating an event to a neighbouring room, the state of some surfaces will be forced to remain constant; i.e. their $T_{j[side]}$ and $q_{rad-lw,j[side]}$. Additionally, the rest will be variable and used as test values $q_{rad-lw,j[side],new}$. Therefore now, these latter variables must be chosen carefully so that Equation (A9) still holds and here we explain our approach to achieve that.

In order to simplify notation let us call $q_{rad-lw,j[side]}$ as x_k according to some ordinal enumeration of the room surfaces. By x_k^{old} and x_k^{new} , we mean the fixed $q_{rad-lw,j[side]}^{old}$ and variable $q_{rad-lw,j[side]}^{new}$, respectively. The Equation (A9) and the new constraints represent the intersection of hyperplanes in a space of $size(B_n) = K$ dimensions (see Equation (A10) in the new notation).

$$\begin{aligned} x_1 + x_2 + \dots + x_K &= 0 \\ x_{c_1} &= x_{c_1}^{old} \\ &\dots \\ x_{c_C} &= x_{c_C}^{old} \end{aligned} \quad (A10)$$

where the amount C of fixed values are named; $c_1, c_2, \dots, c_C \in \{1, \dots, K\}$. Substituting the constraints into the first equation of (A10) we get:

$$x_{v_1} + x_{v_2} + \dots + x_{v_{K-C}} = -(x_{c_1}^{old} + x_{c_2}^{old} + \dots + x_{c_C}^{old}) = d \quad (A11)$$

where v_1, v_2, \dots are the actual surfaces used as variables during the iteration, the rest remain constant. This represents an hyperplane in a subspace of the original problem, or a projected problem, not crossing the origin. The distance vector from the origin is:

$$\mathbf{d} = \frac{\mathbf{a} \cdot \mathbf{d}}{\|\mathbf{a}\|^2} \quad (A12)$$

$$\|\mathbf{a}\|^2 = K - C$$

Any new set $\{q_{rad-lw,j[side]}^{new}\}$ obtained from Equation (5) compatible with the new set $\{T_{j[side]}\}$ must be constrained to the previous hyper-plane. Using the previously computed values, we create the projected vector $\mathbf{x}^{new} = (x_{v_1}^{new}, x_{v_2}^{new}, \dots, x_{v_{K-C}}^{new})$. Next, the hyperplane (A11) is moved to the origin: $\mathbf{x}^{new} = \mathbf{x}^{new} - \mathbf{d}$. The following $K-C$ -vectors always belong to the translated hyperplane:

$$\begin{aligned} \mathbf{w}_1 &= (-1, 1, 0, 0, \dots, 0) \\ \mathbf{w}_2 &= (-1, 0, 1, 0, \dots, 0) \\ &\dots \\ \mathbf{w}_{(K-C)-1} &= (-1, 0, 0, 0, \dots, 1) \end{aligned} \quad (A13)$$

A Gram-Schmidt orthogonalization $\{\mathbf{O}_1, \mathbf{O}_2, \dots, \mathbf{O}_{K-C}\}$ of (A13) is employed to project \mathbf{x}^{new} onto this hyperplane, as an approximate test point (see

Equation (A14):

$$\mathbf{x}^{new}_{proj} = (\mathbf{x}^{new} \cdot \mathbf{O}_1)\mathbf{O}_1 + (\mathbf{x}^{new} \cdot \mathbf{O}_2)\mathbf{O}_2 + \dots + (\mathbf{x}^{new} \cdot \mathbf{O}_{K-C})\mathbf{O}_{K-C} \quad (\text{A14})$$

Finally the result is untranslated thus obtaining the new test values $q_{rad-lw,j[side]}^{new}$ in the projected subspace compatible with the original constraints.

$$\mathbf{x}^{new} = \mathbf{x}^{new}_{proj} + \mathbf{d} \quad (\text{A15})$$

Appendix 3. LIQSS non-autonomous

Originally the LIQSS Migoni (2010) was devised for autonomous systems of the type:

$$\begin{aligned} \dot{x}_1 &= f_1(x_1(t), x_2(t), u(t)) \approx f_1(q_1(t), q_2(t), u(t)) \\ \dot{x}_2 &= f_2(x_1(t), x_2(t), u(t)) \approx f_2(q_1(t), q_2(t), u(t)) \\ &\dots \end{aligned} \quad (\text{A16})$$

where $u(t)$ is an external forcing function. If LIQSS detects that $\dot{x}_j = 0$ within the actual quantum step then it tries to get an estimate \tilde{q}_j of \hat{q}_j :

$$\dot{x}_j = 0 = f_j(q_1(t^-), q_2(t^-), \dots, \tilde{q}_j, \dots, q_n(t^-), u(t^-)) \quad (\text{A17})$$

where t^- indicates the previous value. The estimate is obtained as:

$$0 = \dot{x}_j(x_j(t)) = \dot{x}_j(q_j(t^-)) + \frac{\partial \dot{x}_j}{\partial x_j}(q_j(t^-)) \cdot \Delta Q_j = \dot{x}_j(q_j(t^-)) + A_{jj} \cdot \Delta Q_j \quad (\text{A18})$$

Therefore if $A_{jj} \neq 0$ then the new quantum should be:

$$\Delta Q_j = \frac{f_j(q_1(t^-), q_2(t^-), \dots, q_j(t^-), \dots, q_n(t^-), u(t^-))}{A_{jj}} \quad (\text{A19})$$

However, for the case of non-autonomous systems:

$$\begin{aligned} \dot{x}_1 &= f_1(x_1(t), x_2(t), u(t), t) \approx f_1(q_1(t), q_2(t), u(t), t) \\ \dot{x}_2 &= f_2(x_1(t), x_2(t), u(t), t) \approx f_2(q_1(t), q_2(t), u(t), t) \\ &\dots \end{aligned} \quad (\text{A20})$$

Equation (A19) does not take into account the effect of the time advance $\Delta t = t_a$ from the current quantized state $q_j(t^-)$ until \tilde{q}_j . In this case the estimation A18 must be changed. As in (A19) it is assumed that all variables are constant but x_j and now also t :

$$\begin{aligned} 0 &= \dot{x}_j(x_1, x_2, \dots, \check{x}_j, \dots, \check{t}) = \dot{x}_j(q_j(t^-), t^-) \\ &+ \frac{\partial \dot{x}_j}{\partial x_j}(q_j(t^-), t^-) \cdot \frac{\partial x_j}{\partial t}(q_j(t^-), t^-) \cdot \Delta t \\ &+ \frac{\partial \dot{x}_j}{\partial t}(q_j(t^-), t^-) \cdot \Delta t \\ 0 &= \dot{x}_j(x_1, x_2, \dots, \check{x}_j, \dots, \check{t}) = \dot{x}_j(q_j(t^-), t^-) + \left. \frac{\partial \dot{x}_j}{\partial t} \right|_{\neq t, x_j}(q_j(t^-), t^-) \cdot t_a \end{aligned} \quad (\text{A21})$$

Calling a_j the derivative $\frac{\partial \dot{x}_j}{\partial t}|_{\neq t, x_j}$ (or partial acceleration) the estimated time advance \tilde{t}_a needed to achieve $\dot{x}_j(t) \approx 0$ is:

$$\tilde{t}_a = -\frac{\dot{x}_j(q_j(t^-), t^-)}{a_j} \quad (\text{A22})$$

therefore, now, the quantum ΔQ_j needed is:

$$\Delta Q_j = \int_0^{\tilde{t}_a} (\dot{x}_j(q_j(t^-), t^-) + a_j \cdot t) dt = \dot{x}_j(q_j(t^-), t^-) \cdot \tilde{t}_a + a_j \cdot \frac{\tilde{t}_a^2}{2} \quad (\text{A23})$$

Appendix 4. Components DEVS model

The DEVS formalism explanation can be found in many good books like Wainer and Mosterman (2010). DEVS is a structure :

$$M = (X, S, Y, \delta_{int}(), \delta_{ext}(), \delta_{con}(), \lambda(), t_a()) \quad (\text{A24})$$

where X is the set of input values, S the set of internal states, Y the set of output values and the rest are functions. Namely, $\delta_{int}(): S \rightarrow S$ is the internal transition function, $\delta_{ext}(): Q \times S \rightarrow S$ the external transition function, $\lambda(): Y \rightarrow S$ the output function and $t_a(): S \rightarrow \mathbb{R}_{0, \infty}^+$ the time advance function.

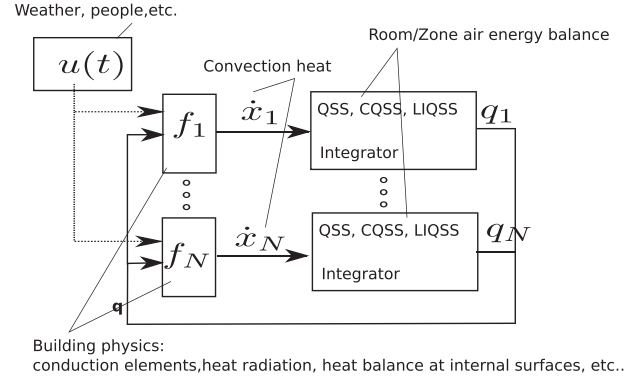


Figure A1. General blocks scheme of a DEVS QSS integration and its mapping to the our DEVS model of Figure 4.

$Q = \{(s, t_e) | s \in S, 0 \leq t_e \leq t_a(s)\}$ is the total state and $\delta_{con}()$ serves to sort how concurrent events are processed. The DEVS simulation engine follows, briefly, the following rules:

- External transition when an external event arrives: 1) $\delta_{ext}()$ is called with t_e less than the last scheduled t_a for the component, 2) a new t_a is scheduled by calling $t_a()$ with the internal state created by the external event.
- Internal transition when next scheduled component needs to be called: 1) the component issues an output by calling $\lambda()$, 2) then there is an internal state change by $\delta_{int}()$, 3) finally $t_a()$ is called and a new time advance t_a is scheduled, i.e. when the component needs to be called again.

The detailed exposition of each DEVS model would require specifying all these structures for each component and falls out of the scope and extent of the paper. Nevertheless here we include some key aspects of the models.

Any component which behaves as a function has some behaviour in common: after an external transition, $t_a()$ returns $0[s]$ in order to make an internal transition instantaneously and after $\lambda()$ outputs a value and $\delta_{int}()$ is called, the $t_a()$ returns $t_a = \infty$, i.e. it remains waiting forever until a new external event arrives.

Appendix 5. Inclusion of the QSS integrator in our building model

Figure A1 shows a very simplified scheme (see the details in Migoni 2010) of a QSS integrator of an ODEs. The idea of a QSS integrator is the following. The functions f_j are used to evaluate the time derivative of each dynamical variable. These values are handed over to the integrator. In the simplest case, the QSS takes that derivative as constant and accordingly, inside the integrator, a continuous variable must evolve linearly. QSS computes the time left for this internal state variable to reach its next quantum value and the DEVS engine schedules the next output event. This is done for all the variables asynchronously. Therefore, the next output event for the QSS integrator-block component will occur for that quantized variable $\{q_1, \dots, q_N\}$ corresponding to the internal variable $\{x_1, \dots, x_N\}$ closer, in time, to its next quantized value. If this happens then all the functions f_j (time derivatives) must be evaluated and each QSS integrator element must re-schedule its next output event. Notice that from the point of view of the root DEVS simulator, an external input $u(t)$ event could arrive before an output event from the QSS-block thus re-scheduling, once more, the next quantum transition of those $\{x_1, \dots, x_N\}$ which depend on the input event.

Since the functions f_j must be evaluated, that means that the components RAD2WALLs, WALL, RAD and GLOBALBALANCE (used inside these f_j , see Figure 4) behave actually as functions. Details of the WALL and RAD components are similar, although not identical due to implementation issues, to those found in Francés, Escrivá, and Ojer (2014). The GLOBALBALANCE component also changes from Francés, Escrivá, and Ojer (2014) for the same reason, but it is worthwhile to expose here, at least the logic of its $\lambda()$ function.

Algorithm 1 GLOBALBALANCE

```

1: procedure  $\lambda(t)$ 
2:    $OK \leftarrow true$ 
3:   if  $Error_{rad-lw} > TOL_{rad-lw}$  then
4:      $OK \leftarrow false$ 
5:   if ! $OK$  then
6:      $Propagation \leftarrow false$ 
7:     Create a vector with the subset of actual variables  $q_{rad-lw,j[side]}$ 
8:     if  $Error_{rad-lw} < Previous\ Error_{rad-lw}$  then
9:        $Previous\ Error_{rad-lw} \leftarrow Error_{rad-lw}$ 
10:    else
11:       $Propagation \leftarrow true$  ▷ Cannot reach a local solution
12:    if ( $OK \ \& \ Propagation$ ) || ( $OK \ \& \ All\ WALLs\ called$ ) then ▷ End
13:      return Event
14:    if ( $OK \ \& \ !Propagation$ ) || ( $!OK \ \& \ Propagation$ ) then ▷ Locally solved || not locally solved, respectively
15:       $Propagation \leftarrow true$ 
16:      Prepare everything for a new propagation
17:    if ( $!OK \ \& \ !Propagation$ ) then ▷ Try to solve locally
18:      Estimate the  $q_{rad-lw,j[side]}^{new}$  using Appendix 2
    return Event

```

Note: The GLOBALBALANCE return value of the *Event* in (Algorithm 1) is not important because some internal information structures are used to pass the information.

Finally, the ROOM/ZONE components are different, they are not functions and contain the QSS integrators (all of them could be just QSS or CQSS or LIQSS, in our case). These are the balance or conservation equations.