4th International Conference on Exploring Service Science 1.3, 7-8 February 2013, Porto, Portugal, LNBIP 143

The final publication is available at link.springer.com

A Service Oriented Simulation Architecture for Intelligent Building Management

Monica Drăgoicea[1], Laurenţiu Bucur[2] and Monica Pătraşcu[1]

[1] University Politehnica of Bucharest
Dept. of Automatic Control and Systems Engineering
313 Splaiul Independenţei, 006042-Bucharest, Romania
monica.dragoicea@acse.pub.ro,monica.patrascu@acse.pub.ro
www home page: http://acse.pub.ro
[2] University Politehnica of Bucharest
Dept. of Computer Science and Engineering
313 Splaiul Independenţei, 006042-Bucharest, Romania
laur.bucur@gmail.com

Abstract. This paper introduces a proposal on developing a service oriented modelling and simulation architecture related to intelligent building management based on an existing open platform that is intended to allow different people to participate and contribute at developing a intelligent building management service ecosystem. In this way, both users and developers can compose new services, while the developer can focus on the most effective use of devices and data, instead of getting lost in upgrading to the latest device driver. The novelty of the proposed framework is the integration in the simulation loop of a Smart Building Controller that can control the real and virtual devices at the facility level. Therefore, the simulation model of the smart building integrates both real as well as simulated devices. The web-based service oriented software application allows to *simulate* the *device-level* and *facility level* behaviour of an intelligent building. It demonstrates a strategy to define an Intelligent Building Management solution that includes scenario simulation, testing of the functionality of the Smart Building Controller (SBC), device monitoring and control, report generation, and implementation of device web services.

Key words: intelligent building management, modelling and simulation, web services, service composition, service ecosystem

1 Introduction

Today, novel modelling and simulation techniques reveal a completely new perspective on evaluating Intelligent Building Management solutions. State-of-the-art perspective on available Smart Building Simulation reveals the following most specific development directions.

Relating to Smart Building Simulation, in [1] the **Performance Framework Tool** suite for simulating a smart building's energy performance is described,

as a set of applications that interoperate and exchange **Building Information Model** data. It is shown that the existing state of the art focuses on the **B**uilding **I**nformation **M**odel 's **I**ndustry **F**oundation **C**lasses (**IFC**) standard (see [2] and [3]), and that in the building lifecycle a great amount of information is lost at various design phases because the **BIM** model is not used consistently. In [1] the proposed PFT tool suite addresses these shortcomings by using the BIM IFC model for simulation and virtual reality visualization with multi–user "avatar" capabilities in the Pudecas Smart Building Simulation and Visualisation software.

In [4] the Interactive Smart Home System (ISS) is described, as an interactive smart home simulator. In this research, the smart house is considered as being an environment made up of independent and distributed devices interacting to support user's goals and tasks. Therefore, by using ISS, the developer can realize the relationships between virtual home space, surrounded environment, users and home appliances.

Smart Energy Solutions' Smart Energy simulator [5] is a cloud-based simulator for smart buildings. Its main features refers to energy analysis and thermal load simulation (Smart Energy web-application). The Smart Energy application includes a powerful thermal simulation engine that facilitates the web based simulation, it requires no installation, it includes the building model in simulation, schedule simulation, air flow simulation, occupant-based analysis, predefined inputs based on real-world data (temperature), and user friendly visual reports.

Other approaches are oriented towards the adaptation of computer games and related technologies to create virtual and mixed reality intelligent environments (an extended discussion can be found in [6]).

Along with ensuring comfortable and green living environments, special attention should be given to sustaining structural integrity. The main danger to buildings' lifespan is caused by earthquakes and other natural disasters that can interact negatively with the purpose of Smart and/or Intelligent Buildings. Researche of the past decades reveals new damping technologies (see for example [7],[8]) and control strategies (see for example [9],[10],[11],[12]) that consistently ensure structural safety prior, during and post seismic activity.

This present research proposes a service-oriented approach of creating and using simulation models based on real and virtual devices integrated in a building model in order to a) test the functionality of intelligent building control components (like smart building controllers) and b) to estimate and evaluate power consumption at the facility level. The real and virtual device models are hosted and exposed by a Simulation Engine web service and optionally by a Simulation Bridge web service. The fundamental advantage of this approach over related references (see [13], for example) is a unitary description of the devices based on web services and their integration in a Service Oriented Architecture for Intelligent Building Management (section 2). These device web services can be composed with a Smart Building Controller in order to test policies and schedules (section 4), without the need of a dedicated building infrastructure. In addition, the proposed approach integrates both human and structural safety

(seismic events and response to natural disasters are considered), making the architecture extensible and flexible. Thus, considering its pertinence to a large range of buildings (regardless of infrastructure), its ability to incorporate a variety of devices (from elevator controllers and ventilation systems to seismic dampers and fire protection), the approach that is discussed in this paper is extensive and has a wide applicability expanse.

Therefore, in this proposed framework a *service ecosystem* dedicated to the Intelligent Building Management solution development could be eventually created, both for *control* and *monitoring*, as well as for *utilities cost estimation*. Such a service ecosystem would comprise a group of users within the industry *sharing* their *services* for use across the group for *testing* or *training*.

This paper is organized as follows. Section 2 presents a novel service oriented perspective on developing Intelligent Building Management solutions. Section 3 introduces the main contribution of this work, the proposed service oriented framework for smart building modelling and simulation that can integrate virtual device models (through web services) and real devices. Section 4 presents working results of the proposed strategy that uses simulation models integrating virtual device models and real devices for testing the SBC behaviour. Section 5 presents conclusions and further directions of development.

2 A Perspective on Service Orientation for Intelligent Building Management

This section shortly describes the service-oriented approach to control and manage building facilities via intelligent controllers and a web-based portal [14] that consists on an infrastructure for people to share their building control services using a service-oriented platform (Fig. 1), so that:

1. users could define, search and compose building services according to their facility needs through a portal, **Environment Manager**;
2. users can define working policies and schedules using a **Policy Editor** as explained in section 3;
3. users can execute simulation before deployment accessing through a portal a **Simulation Engine** by help of a **Simulation Console**.

The operation of the whole service-oriented framework is based on specific service composition mechanisms, as follows:

1. Composition between the web-based building control application (**Environment Manager**) and a **Smart Building Controller** (WCF web service). The Environment Manager consumes the Smart Building Controller (SBC) service to enumerate devices, classes of devices, schedules and operation timetables. At the same time, it allows to visualize the state of the devices in the SBC, their direct control, as well as to edit the SBC's schedules and operation timetables;

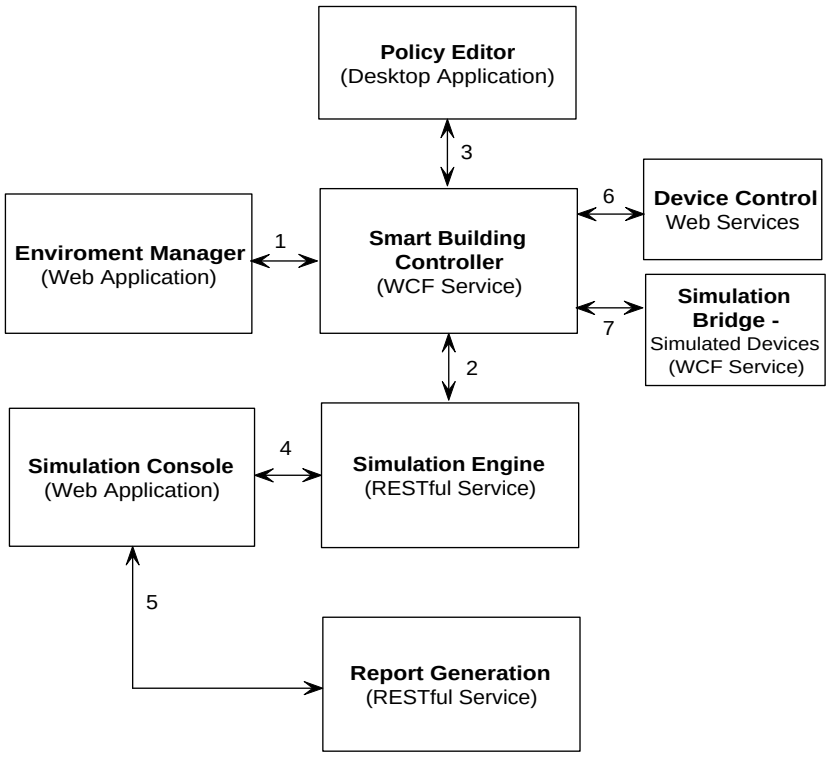

Fig. 1. The Service-Oriented Approach for Intelligent Building Management

2. Composition between a **Smart Building Controller** (WCF web service) and the **Simulation Engine** (RESTful web service). This mechanism allows to include real devices from the SBC in a simulation, and to test the Smart Building Controller in different operation scenarios. In this respect, the controls and events are sent directly from the simulation scenario to the SBC, to be processed. In this way, the functioning of the devices and the policies defined in the SBC which are triggered by the events can be tested;
3. Composition of a **Policy Editor** (desktop application) with a **Smart Building Controller** (WCF web service). This will allow to manage and edit the functioning policies of the Smart Building Controller;
4. Composition of the **Simulation Console** (web application) with the **Simulation Engine** (RESTful web service). This allows to expose the functionality of the Simulation Engine as a web application to develop simulations;
5. Composition of the **Simulation Console** (web application) with the **Report Generation Service** (RESTful web service). This allows to generate standard (PDF) simulation reports and to estimate power consumption and operation costs of the devices present in a building;

6. Automated composition of a **Smart Building Controller** (WCF web service) with the device control web services. This allows to control real devices by using the corresponding web services and to automatically compose these services with the Smart Building Controller web service;
7. Composition of a **Smart Building Controller** (WCF web service) with the virtual device control web services, through the simulated device control web service, i.e. the **Simulation Bridge** (WCF web service) [15].

3 A Service-Oriented Framework for Smart Building Simulation

The proposed Simulation Framework is based on a *service-oriented, web-based* **Smart Building Simulator** (**SBS**). In fact, the SBS is obtained by the *composition* of a **Simulation Console** (web application) with a **Simulation Engine** (RESTful web service), see branch 4 in Fig. 1. This allows to expose the functionality of the Simulation Engine as a web application to develop simulation scenarios.

Further, the **Simulation Engine** (RESTful web service) can be *composed* with the **Smart Building Controller** (web service) and this composition will allow to include in the simulation scenario the real devices controlled by the SBC, and virtual device models through the **Simulation Bridge**, in order to test the Smart Building Controller in different scenarios (see branch 2 in Fig. 1 and section 4 for a detailed example).

The two main goals of the Simulation Framework are the following:

1. *testing* the response of a Smart Building Controller (SBC) in various simulation scenarios (see section 4 for example and simulation results);
2. *estimating* the utility bill for a real building or a virtual building (whose model integrates virtual device models) based on the definition, composition and simulation of a simulation model.

The Simulation Framework Architecture that integrates the Smart Building Simulator (SBS) is outlined in Fig. 2.

The **Simulation Console** communicates with the **Simulation Engine** to create, manage and execute *smart building simulations*. The **Simulation Engine** manages one or more *simulation scenarios* in a simulation [16].

On the perspective of the Simulation Framework proposed in this paper, we use the following definition of a **simulation scenario**:

Definition 1 A *simulation scenario* is a perspective created in the Simulation Console by the composition of the following elements:

1. a set of time ordered *events* that are processed by the Simulation Engine and visualised in the Simulation Console in the timeline of the simulation scenario. They can be exchanged between the Simulation Engine and the Smart Building Controller, in both directions, such as: a) *External events*

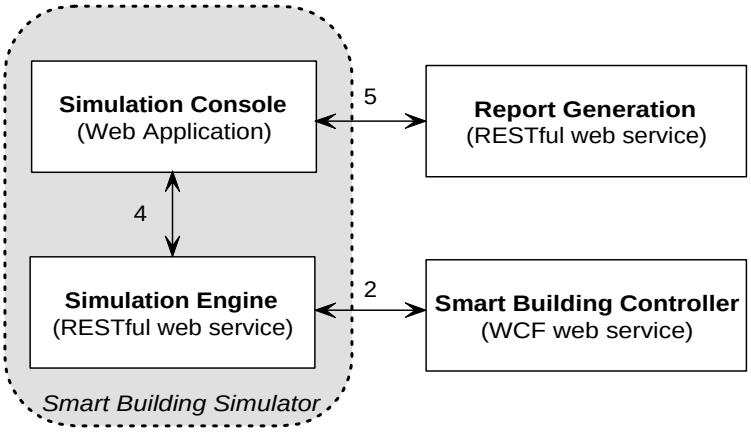

Fig. 2. The Smart Building Simulator in the Service-Oriented Simulation Framework

can be sent from the simulation scenario (by the Simulation Engine) to be processed by the Smart Building Controller, and b) The events from the SBC can be imported into the simulation, instantiated and visualized through the `EventTypes` window in Simulation Console;
2. a set of *virtual devices*, that means virtual objects hosted, created and managed in the Simulation Engine, as instances of virtual device classes defined by the user for a given simulation. Virtual devices hosted by the Simulation Engine are used for energy estimation only;
3. a set of *real devices*. The real devices are the devices installed at the facility level. They can be recognized and **controlled** by the Smart Building Controller through the corresponding web services. Real device models imported from the SBC in the simulation console are used by the Simulation Engine to send commands to real devices during simulation.

The Simulation Framework allows the user to define one or more simulation scenarios that can include three different types of *events*:

1. *Control Command* events. The *CommandEvent* is related to the `Start/Stop` operations for virtual and real devices. These commands are used to start and stop devices and are mainly used during energy consumption simulation;
2. *Change Of Value* (CoV) events. The *ChangeOfValue* event can be used to change the value of a device property in the simulation. The user may choose to place a *ChangeOfValue* event on the simulation scenarios timeline that changes the power consumption or any parameter of a device to a specific value. These events are related to virtual and real devices in a simulation. In case of virtual devices the events are executed internally by the Simulation Engine while for real devices the events are routed and executed to the SBC, thus **controlling** real devices during simulation;

3. *External events*. They are asynchronous user-defined event types that are defined in and recognized by the Smart Building Controller as XML serializable object types. Through the Simulation Console, the Simulation Engine imports the definitions of all the external event types defined in and recognized by the Smart Building Controller. External events can be instantiated on the simulation timeline of the simulation scenario. When such an event is executed during simulation, its XML serialized form is sent to the SBC's event queue.

When the execution of the simulation reaches one of the control command, change of value or external event, the Simulation Engine (RESTful web service) either executes the event internally or it is composed with the Smart Building Controller (WCF web service) and routes events to the SBC. This is summarized in Fig. 3.

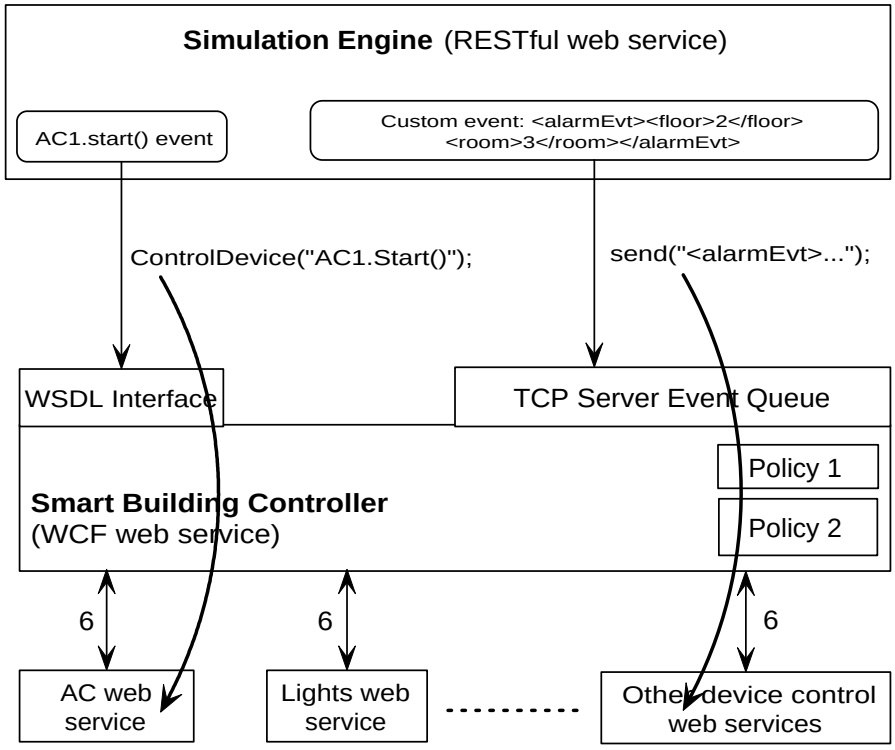

Fig. 3. Composition of the Simulation Engine and the Smart Building Controller

4 Testing the response of a Smart Building Controller

The Smart Building Controller is the client-side service orchestration component of the Service-Oriented Architecture [14]. Its role is to manage the operation of physical building devices that are controlled by web services. The SBC, as an orchestrator, manages the execution of device control web services by different sets of user-defined operating *schedules* and *policies* [17].

Specific *policies* apply for the real-time behaviour of the Smart Building Controller, using a specific set of rules combining *event triggers*, *conditions* and *actions* to be executed if certain conditions are fulfilled or exceptions occur.

Definition 2 A Smart Building Controller *policy* is a set of rules of the form:

ON (Trigger) **IF** (Condition) **THEN** (Execute block of actions)
ELSE (Execute block of actions)

where:

- *Trigger* specifies the name of the rule trigger. It can be set to `true` or it can take the name of an event type registered in the Smart Building Controller, such as `EarthquakeEvent` (see section 4.1);
- *Condition* is an expression recognized by the Smart Building Controller at runtime and evaluated as a `boolean` value.

Note: A condition can be composed from a set of web service call expressions, device property reads, using a special Smart Building Controller calling convention.

The behaviour of the Smart Building Controller can be tested in various scenarios. A Smart Building Controller test scenario may include real, as well as virtual devices through the Simulation Bridge. Testing the Smart Building Controller operation implies two aspects:

1. Sending *ChangeOfValue* events for various *real devices* to the Smart Building Controller, as one or more web service calls and observing the response of the Smart Building Controller in the real environment (the event is executed and the device is controlled by altering its operating parameters);
2. Sending asynchronous events (external events in simulation) to the SBC's event queue and observing the results of the Smart Building Controller executing the event-triggered policy in the real environment.

In this case, the composition between the Simulation Engine and Smart Building Controller, through the Simulation Console, is performed as follows:

1. The communication from the Simulation Engine to the Smart Building Controller consists of:
   a) Establishing the connection to the SBC and create a blank simulation scenario in the Simulation Console;
   b) Sending external events defined in the simulation scenario (in the Simulation Console) to the SBC;

   c) Listing available devices defined in the SBC in the Simulation Console;
   d) Sending *ChangeOfValue* commands to the devices installed in the SBC.
2. The communication from the Smart Building Controller to the Simulation Engine consists of:
   a) Importing the events defined in the SBC in the Simulation Console. The SBC is ready to receive events through the external events mechanism of the Simulation Engine;
   b) Visualizing the event types in the `EventTypes` window in the Simulation Console.

In order to test the behaviour of the Smart Building Controller, the following steps should be fulfilled in order to define and execute the corresponding simulation scenario:

1. Define the events in the simulation scenario to be sent to the Smart Building Controller. They are defined as external events and correspond to the events list defined in the SBC (section 4.1);
2. Define the simulation scenario in the Simulation Console. It may include real devices in the building infrastructure recognized and controlled by the SBC and models of the virtual devices through the Simulation Bridge (section 4.2);
3. Define the policies and schedules for the Smart Building Controller operation, through the Policy Editor (section 4.3);
4. Execute simulation scenario in the Simulation Console and inspect the SBC's execution logs (section 4.4).

The following subsections present results that concern testing the SBC operation in a earthquake scenario using models of dedicated virtual devices through the web service definition mechanism in the Simulation Bridge. Previous work on this direction is related to the definition of the way in which emergency response protocols can be combined at a microscopic level with a Smart Building Controller such as a high level of performance in what concerns comfort could be assured [18].

4.1 Testing: Define a set of events

We define an `EarthquakeEvent` as a SBC serializable class containing the following fields:

1. `Magnitude_min`: a double representing the minimum estimated magnitude of the earthquake;
2. `Time_to_earthquake`: is a number (double) expressing the estimated number of seconds until the earthquake actually occurs (see for reference [19]);
3. `Geographical_area`: specifies the area in which the earthquake will occur. For example, *Vrancea*.

The event type is registered in the Smart Building Controller. This means compiling the class in a shared .NET assembly and placing it in the `SBC EventTypes` folder. At startup, the Smart Building Controller will register this event by enumerating all public serializable classes from the `EventTypes` folder shared assemblies.

4.2 Testing: Define a simulation scenario

The simulation scenario for the earthquake event management is presented in Fig. 4. A blank simulation is created in the Simulation Console by connecting to the SBC. A simulation scenario is defined spanning one day. The list of events recognized by the SBC is automatically imported in the simulation model.

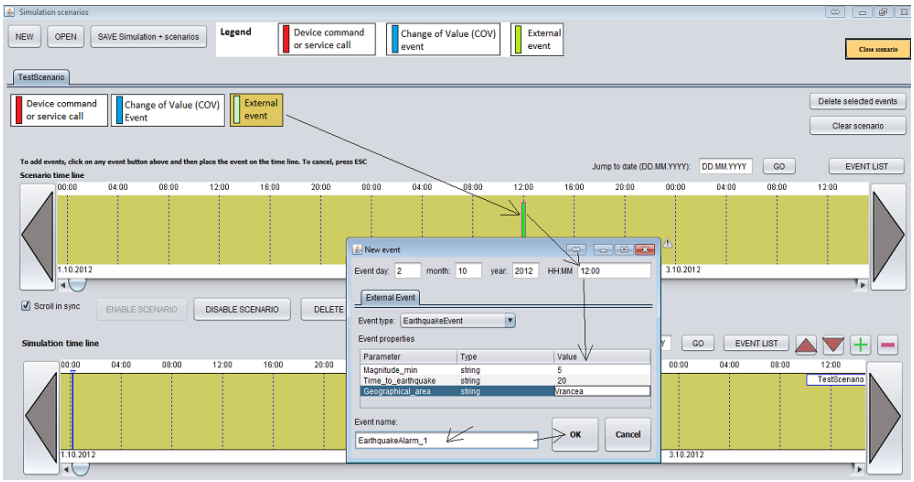

Fig. 4. Simulation scenario - Smart Building Controller earthquake testing scenario

The Simulation Bridge [20] is started and a virtual building device configuration is created with the following components (see also the connection configuration in Fig. 5):

1. `GasSwitch`: a virtual device that controls the gas distribution in the building exposed with a single parameter: state = 0 ot 1
2. `Light1`, `Light2`, `Light3`: 3 virtual lights, exposed with a single parameter: state = 0 or 1
3. `EarthquakeAlarm`: a virtual earthquake acousting alarm exposed with a single parameter: running = 0 or 1
4. It is assumed the building is situated in `Vrancea` geographical area or Romania.

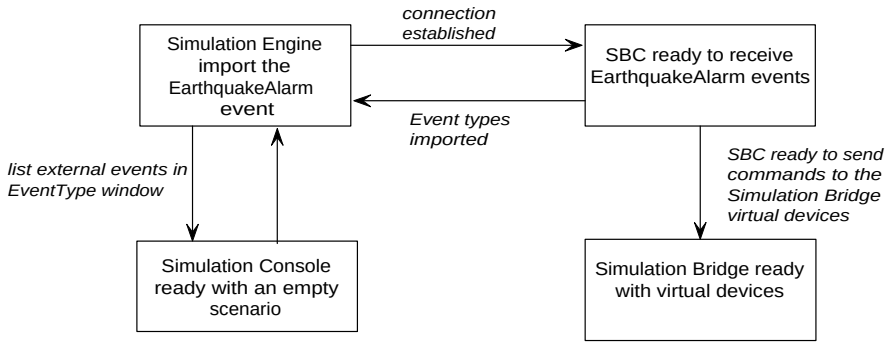

Fig. 5. List of connections - Smart Building Controller earthquake testing scenario

4.3 Testing: Define a set of policies

A Critical System Emergency Protocol may incorporates two policies: the seismic response policy (SRP) and the disaster response policy (DRP) (see a detailed discussion in [18]). Low level and high level policies could be defined, as follows:

– a low level policy deals with executive level actions. For example:

If earthquake magnitude is higher than 4, then turn on the active damping system. Else, maintain the passive damping system.

– a high level policy queries one or more facility management systems and executes one or more low level policies. For example:

If earthquake alarm active and occupancy > 40% and security understaffed, then apply emergency evacuation protocol number 6.

A three rule policy named `EarthquakeAlarmPolicy` is defined using the Policy Management System [21] as follows:

Rule 1 : If there are at least 10 seconds until the earthquake hits Vrancea, start the acoustic alarm:

ON(EarthquakeEvent) IF (message.Geographical_area.Equals(Vrancea) and (message.Time_to_earthquake >=10) THEN [[EarthquakeAlarm.running=1]];

Rule 2 : If there is an earthquake in this area, if the magnitude is greater than 4, turn off the lights:

ON(EarthquakeEvent) IF (message.Geographical_area.Equals(Vrancea) and message.Magnitude_min >=4.0) THEN [[GasSwitch.state=0]];

Rule 3 : No matter how large the magnitude, if the earthquake hits Vrancea, turn off the three lights:

ON(EarthquakeEvent) IF (message.Geographical_area.Equals(Vrancea) THEN [[Light1.state=0]]; [[Light2.state=0]]; [[Light3.state=0]];

4.4 Testing: Execute simulation scenario

The goal of this step was to test this policy in 3 earthquake alarm scenarios:

Behaviour 1 : On a magnitude 5 earthquake hitting Vrancea within the next 20 seconds, this policy would activate the virtual acoustics alarm, stop all lights and turn off the gas distribution installation.

Behaviour 2 : On a magnitude 3 earthquake alarm hitting Vrancea in the next 5 seconds, this policy would turn off the Lights.

Behaviour 3 : On a magnitude 4 earthquake alarm hitting Vrancea in the next 8 seconds, this policy would turn off the gas installation and the lights.

Three custom events of the type `EarthquakeEvent` were created on the simulation time line, with the following fields (time expresses in seconds):

Event 1 : At simulation time 12:00 – Magnitude_min = 5, Time_to_earthquake = 20, Geographical_area =Vrancea. Desired behaviour: **Behaviour 1**

Event 2 : At simulation time 14:00 – Magnitude_min = 3, Time_to_earthquake = 5, Geographica_area = Vrancea. Desired behaviour: **Behaviour 2**.

Event 3 : At simulation time 16:00 – Magnitude_min = 4, Time_to_earthquake = 8, Geographical_area = Vrancea. Desired behaviour: **Behaviour 3**.

An excerpt of the obtained results are present in the following figures and tables.

Table 1 depicts the policy execution in the SBC and the event log when the event 1 is received in the SBC.

Table 1. Events log in the Smart Building Controller - policy execution

Date	Time	Device ID	Logged actions	Type
14.09.2012	00:31:37.545	0	EarthquakeEvent: Magnitude_min=5 Time_to_earthquake=20 Geographical_area=Vrancea	event
14.09.2012	00:31:37.590	1211008	EarthquakeAlarm: running=1	Earthquake Alarm Policy
14.09.2012	00:31:37.608	1211004	GasSwitch: state=0	Earthquake Alarm Policy
14.09.2012	00:31:37.617	1211005	Light1: state=0	Earthquake Alarm Policy
14.09.2012	00:31:37.626	1211006	Light2: state=0	Earthquake Alarm Policy
14.09.2012	00:31:37.638	1211007	Light3: state=0	Earthquake Alarm Policy

The earthquake event `Event 1` is sent by the Simulation Engine, a Single Step simulation is executed and the event is correctly received by the SBC (Fig. 6).

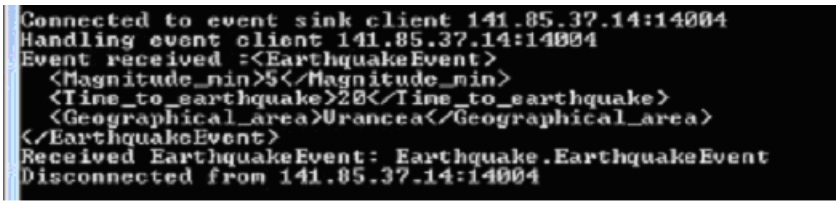

Fig. 6. The earthquake event is received in the SBC

The Event Manager window depicts the SBC log (Fig. 7).

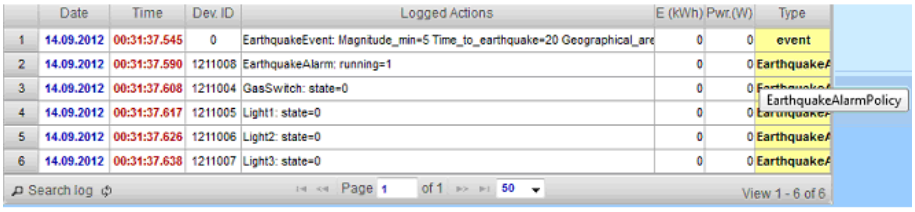

Fig. 7. The earthquake event is received in the SBC - SBC log

5 Conclusions and further development directions

The Simulation Framework introduced in this paper serves two purposes. On one hand, it assist in testing the behaviour of a Smart Building Controller using simulation scenarios. This is achieved by creating a simulation scenario containing events to be sent to and processed by a real Smart Building Controller. On the second, it may further allow estimating the energy consumption in a building environment without real devices being installed. This option will be developed in future experiments. Optionally, the simulator can also estimate the energy consumption of real devices by creating a simulation from the set of devices installed in a Smart Building Controller.

Apart from the already mentioned advantages of this solution, this work introduces the notion of *policy*, that means a set of rules that govern device control according to user preferences. The novelty of this work consists on the extension of the service oriented mechanisms at the policy level, that means the definition of the conditions for the Smart Building Controller policies based on web service composition and testing their proper execution using a service-oriented simulation framework. Due to the service oriented nature of the Smart

Building Controller, a condition can be composed from a set of web service call expressions, device property reads, using a special Smart Building Controller calling convention.

Further developments of the proposed service oriented infrastructure would approach an openly integration of functionalities, devices, simulation and operation scenarios, related to different operation and security aspects for intelligent building management. An example referring to earthquake monitoring and response is included that depicts testing the behaviour of the Smart Building Controller in simulation scenarios that include virtual devices control.

Also, this strategy can be used to provide users with practical and research experience with HITL (hardware-in-the-loop) technique for training and testing of Intelligent Building Management solutions.

Acknowledgments. This work was supported by INSEED - Strategic Grant POSDRU No. 57748 (2010), co-financed by the European Social Fund – Investing in People, within the Sectoral Operational Programme Human Resource Development 2007 – 2013.

References

1. McGlinn, K., Corry, E., O'Neill, E., Keane, M., Lewis, D., O'Sullivan, D.: Monitoring Smart Building Performance Using Simulation and Visualisation. In: Proceedings of the Ubicomp 2010 Workshop - Ubiquitous Computing for Sustainable Energy, UCSE2010, pp. 41-47 (2010)
2. International Alliance for Interoperability. Industry Foundation Classes. Available at http://www.buildingsmart.com/, accessed April, 2009
3. InfoComm BIM Taskforce. Building Information Modelling. Available at http://www.infocomm.org/cps/rde/xbcr/infocomm/Brochure_BIM.pdf, accessed April, 2009
4. Van Nguyen, T., Kim, J.G., Choi, D.: ISS – The Interactive Smart Home Simulator. In: 11th International Conference on Advanced Communication Technology, (ICACT), pp. 1828-1833, (2009)
5. Smart Energy, Web-based Energy Modelling Software. Available on-line at http://www.smartenergysoftware.com, accessed April 20 (2009)
6. Davies, M., Callaghan, V.: Towards Producing Artificial Humans for Intelligent Environments Research. In: Intelligent Environments 2011 (IE11), Nottingham 27-29th, July (2011)
7. Ma, H., Yam, M.C.H.: Modelling of a self-centring damper and its application in structural control. In: Journal of Constructional Steel Research, (JCSR), vol. 67, pp. 656-666, (2011)
8. Bitaraf, M., Ozbulut, O.E., Hurlebaus, S., Barroso, L.: Application of semi-active control strategies for seismic protection of buildings with MR dampers. In: Engineering Structures, (ES), vol. 32, pp. 3040-3047, (2010)
9. Ali, Sk.F., Ramaswamy, A.: Optimal fuzzy logic control for MDOF structural systems using evolutionary algorithms. In: Engineering Applications of Artificial Intelligence, vol. 22, pp. 407-419, (2009)

10. Oates, W.S., Smith, R.C.: Nonlinear Optimal Control Techniques for Vibration Attenuation Using Magnetostrictive Actuators. In: Journal of Intelligent Material Systems and Structures, (JIMSS), vol. 19, pp. 193-209, (2008)
11. Shook, D.A., Roschke, P.N., Lin, P.-Y., Loh, C.-H.: GA-optimized fuzzy logic control of a large-scale building for seismic loads. In: Engineering Structures, (ES), vol. 30, pp. 436-449, (2008)
12. Neelakantan, V.A., Washington, G.N.: Vibration Control of Structural Systems using MR dampers and a 'Modified' Sliding Mode Control Technique. In: Journal of Intelligent Material Systems and Structures, (JIMSS), vol. 19, pp. 211-223, (2008)
13. Virtual Device Technologies. Techniques for Developing Virtual Device Models. Available on-line at http://virtualdevicetech.com/technologies/hdvd.php, accessed at 12 September, 2012
14. Tsai, W.T., Petrescu, S., Bucur, L., Chera, C.: A Service-Oriented Approach for Intelligent Building Management. In: 18th International Conference on Control Systems and Computer Science, CSCS 18, pp. 676-681, vol. 2, ISSN 2066-4451, Bucharest, Romania (2011)
15. FCINT Specification for Web Services. Available on-line http://www.fcint.ro/portal/Documents/
16. FCINT Simulator Tutorial. Available on-line at http://portal.fcint.ro/Simulator/Simulator.html
17. Bucur, L., Tsai, W.T., Petrescu, S., Chera, C., Moldoveanu, F.: A Service-Oriented Controller for Intelligent Building Management. In: 18th International Conference on Control Systems and Computer Science, CSCS 18, pp. 665-670, vol. 2, ISSN 2066-4451, Bucharest, Romania (2011)
18. Drăgoicea, M., Pătraşcu, M., Bucur, L.: Service Orientation for Intelligent Building Management: An IoS an IoT Perspective. In: UNITE 2nd Doctoral Symposium R & D in Future Internet and Enterprise Interoperability. 11-12 October, Sofia, Bulgaria (2012)
19. NIEP National Institute for Earth Physics, Romania. Seismic rapid early warning system for dangerous facilities. Available on-line at http://www.infp.ro/real-time/ews
20. FCINT Simulation Bridge. Available at http://www.fcint.ro/portal/Downloads/FCINT_SimulationBridge.zip/
21. FCINT Policy Management System. Available for download on-line at http://www.fcint.ro/portal/Downloads/FCINT_PolicyManagementSystem_kit.zip/