# SFIDE: A SIMULATION INFRASTRUCTURE FOR DATA CENTERS

Ignacio Penas

Green Integrated Systems Laboratory
Technical University of Madrid
Av. Complutense, 30
28040, Madrid, Spain
nacho@die.upm.es

Marina Zapater

Dpt. of Computer Architecture and Automation
Complutense University of Madrid
C/Prof. José García Santesmases, 9
28040, Madrid, Spain
marinzap@ucm.es

José L. Risco-Martín

Dpt. of Computer Architecture and Automation
Complutense University of Madrid
C/Prof. José García Santesmases, 9
28040, Madrid, Spain
jlrisc@ucm.es

José L. Ayala

Dpt. of Computer Architecture and Automation
Complutense University of Madrid
C/Prof. José García Santesmases, 9
28040, Madrid, Spain
jayala@ucm.es

## ABSTRACT

Data centers are huge power consumers and have very high operational costs. Both industry and academia have proposed strategies at multiple levels (server, room layout, cooling, workload allocation, etc.) to increase the efficiency of these facilities. Testing the impact of variables so different in nature such as layout or workload allocation can only be managed by simulators. Current simulation infrastructures are either focused on data room thermal dynamics, or target only a specific stage of data center operations, such as workload allocation. Moreover, they are geared for specific use-cases such as HPC or cloud computing. In this paper we present a data center modeling and simulation framework, for both HPC and cloud applications, to assess data center performance, thermal behavior, energy efficiency and operational cost. Our goal is to show the possibilities of the current data center modeling and simulation framework. Furthermore, as we provide a fully configurable, flexible and scalable infrastructure any kind of policy, data center size or workload amount could easily be implemented over the simulator. We also provide the data sets used to validate our models and policies, obtained from real servers and data centers, so as to enable researchers to test their strategies in a realistic setup.

**Keywords:** DEVS, Data Center Simulation, Optimization, Smart Grid.

## 1 INTRODUCTION AND RELATED WORK

Nowadays, data center contribution to European electricity consumption is estimated to be between 2 and 2.5%, with an annual growth from 10 to 15% (Engbers and Taen 2014). The growing popularity of Cloud computing, together with the development of next-generation applications such as e-Health, and the explosion of the Internet of Things (IoT) has increased exponentially the services provided, generating a huge

*SummerSim-DEVS, 2017 July 9-12, Bellevue, WA, USA; ©2017 Society for Modeling & Simulation International (SCS)*

amount of data that needs to be processed, analyzed and provided to users in a meaningful way. The unsustained energy consumption of these facilities, together with the proliferation and growth of data centers represents an important challenge and is the top concern for reaching exascale computing (DOE ASCAC Subcommittee 2014).

Traditionally, data centers have been roughly divided between Cloud computing and High Performance Computing (HPC) infrastructures. In Cloud computing facilities we can find traditional virtualized applications, i.e. Virtual Machine (VM) hosting, together with scale-out applications, i.e. data serving. Traditional cloud applications have low computational demand and enhance energy efficiency by sharing resources among users, i.e. increasing server utilization by using virtualization and consolidation techniques. Scale-out applications, on the contrary, are latency-critical or memory bounded applications that handle independent requests distributed across servers, and need to maximize per-server throughput. Finally, HPC applications need to deal efficiently with computationally-intensive tasks, i.e., CPU and memory bounded applications. These can be split in parameter-sweep applications that run many instances of independent jobs, and data analytics and number crunching jobs in which data communication and sharing across threads cannot be neglected.

The evolution of data centers is highly constrained to the current computing paradigm. Next-generation applications need to deal efficiently with computationally-intensive tasks to process and analyze the retrieved data, but must also provide efficient Cloud platforms. Regardless of the application being executed, data centers must provide dynamic and flexible services, offering guaranteed performance or strict Quality of Service (QoS) to users. And both HPC and Cloud applications need to provide the best performance with the lower energy and operational cost possible. For years, energy reduction has focused on maximizing data center efficiency, usually computed using the Power Usage Effectiveness (PUE) metric, defined as the ratio between total facility power and IT power. In year 2013, world average PUE reached 1.65 (Matt Stansberry 2013), whereas some major players such as Google are already reporting PUE values of around 1.13 (Inc. 2017).

Even in highly-efficient next-generation data centers, with low PUE values and reduced cooling waste, power density and energy consumption remain to be an important challenge. To further improve efficiency, there is a need to develop new data center models and optimization policies. Gathering the required information to train and test models is complex. Adopting untested policies and algorithms in a real Data Center is almost impossible. Given these constraints, simulation has become a great alternative to assess the performance of data centers in terms of power, energy, performance, quality of service, operational costs, etc. Optimizing decision variables so different in nature such as the data center physical layout or the benchmarking of applications under variable conditions can only be managed by simulators. However, to offer a useful tool for researchers, companies and data center operators, data center simulators need to provide methods to incorporate accurate models tested on real equipment, as well as to test new optimization policies. This imposes the need for plugin-based simulators able to evolve as new cooling techniques and IT equipment are introduced.

Current Data Centers simulators have not been designed to tackle the previous challenges. From the industry perspective, simulation has traditionally focused on predicting the thermodynamics of data centers at design time using Computational Fluid Dynamics (CFD) (Singh, Singh, Parvez, and Sivasubramaniam 2010). The high economic cost of CFD software prevents re-running simulations to test changes such as the acquisition of new equipment. In order to assess energy optimization policies, researchers have started to develop their own simulation frameworks. However, these simulators tackle only thermal dynamics, or are application-specific. On the one hand, simulators such as SimWare (Yeo and Lee 2012) or CoolSim (Software 2016) focus only on the data room, disregarding performance, scheduling, allocation, and operational costs. Some application-specific simulators focus on Cloud computing scenarios (such as Cloudsim (Calheiros, Ranjan, Beloglazov, De Rose, and Buyya 2011)), raising their abstraction level to the virtual machine and disregarding the data center room-level aspects such as layout, cooling, heat recirculation, etc. Others only focus on the performance obtained via the scheduling and allocation of HPC tasks (such as the BSC Slurm Simulator (Lucero 2011), or the SST simulator (Rodrigues, Bergman, Bunde, Cooper-Balis, Ferreira, Hemmert, Barrett, Versaggi, Hendry, Jacob, Kim, Leung, Levenhagen, Rasquinha, Riesen, Rosenfeld, Del Carmen Ruiz Varela, and Yalamanchili 2012)). A common drawback of these tools is the lack of well-defined interfaces, the inherent difficulty in configuration, and the orientation to software engineers. Even though some simulators are widely spread in the community, they do not provide ways to easily configure the simulator, and require the researcher or data center operator to develop new code to launch simulations. Among simulators, perhaps iCanCloud (Núñez, Vázquez-Poletti, Caminero, Castañé, Carretero, and Llorente 2012) is the one providing easier configuration tools, as it is a library for the OMNeT++ framework (Varga 2001). The creators of Cloudsim have also recently released a new version of their simulator, called CloudNetSim++ (Malik, Bilal, Malik, Anwar, Aziz, Kliazovich, Ghani, Khan, and Buyya 2015), which also uses OMNet++, to tackle usability issues and include networking using the INET framework. However, in both cases, the simulation engine of these tools is completely coupled with the models provided, making it difficult to incorporate new models and functionality into the simulator.

Our work proposes a data center simulation framework, named SFIDE (Simulation Framework and Infrastructure for Data cEnters), which allows service providers, third-party software developers and researchers to test current or prototyping infrastructures and software packages in terms of performance, thermal efficiency and operational cost. To this end, we need a practical and efficient way of applying Modeling and Simulation (M&S) to the development of the system at an early stage. Moreover, we must separate the models themselves from the simulation platform so that modeling experts can focus on model abstractions. As a result, we have applied the Discrete EVent Systems (DEVS) formalism (Zeigler, Praehofer, and Kim 2000). Table 1 summarizes the main aspects of the above mentioned simulators and compares their features against our proposed simulation framework.

In this paper, we propose a simulation framework for HPC and Cloud applications that enables researchers to incorporate their developed models and optimizations and test the impact at the data center scale under various configurations, without the need to develop new functionality, abstracting them from the simulator internals. We also provide a fully configurable, flexible and scalable simulator infrastructure, specifically tailored for new-generation data centers, where power, temperature and performance models can be easily plugged-in, and resource management policies can be tested in terms of performance and energy. Finally, in order for the research community to be able to work with realistic setups, we also provide the data sets used to validate our models and policies.

The remainder of the paper is organized as follows: Section 2 describes the basic principles of the DEVS formalism. Section 3 provides the details of the model architecture. Section 4 briefly shows how the simulation model works. Finally, the most important conclusions are drawn in Section 5.

## 2 DEVS MODELING AND SIMULATION

DEVS is a general formalism for discrete event system modeling based on set theory (Zeigler, Praehofer, and Kim 2000). The DEVS formalism provides the framework for information modeling which gives several advantages to analyze and design complex systems: completeness, verifiability, extensibility, and maintainability. Once a system is described in terms of the DEVS theory, it can be easily implemented using an existing computational library. After 15 years, the parallel DEVS (PDEVS) approach was introduced as a revision of Classic DEVS. Currently, PDEVS is the prevalent DEVS, implemented in many libraries. In the following, the use of DEVS implies PDEVS.

| | CoolSim | SimWare | BigHouse | SST | CloudSim | iCanCloud | CloudNetSim++ | SFIDE |
|---|---|---|---|---|---|---|---|---|
| Platform | - | - | - | - | - | OMNet++ | OMNet++ | OMNet++/DEVS |
| Language | ? | C++ | Java | C++ | Java | C++ | C++ | C++ |
| Parallel | No | No | No | No | No | Limited | No | Yes (DEVS) |
| Distributed | No | No | No | No | No | No | No | Yes (DEVS) |
| Cloud support | No | No | No | No | Yes | Yes | Yes | Yes |
| HPC support | No | No | Yes | Yes | No | No | No | Yes |
| Performance | No | No | Yes | Yes | No | No | No | Yes |
| Server power | No | No | No | No | Yes | Yes | Yes | Yes |
| Server temperature | No | No | No | No | No | No | No | Yes |
| Networking models | No | No | No | Yes | No | Yes (INET) | Yes (INET) | Yes (INET) |
| Data room dynamics | Yes | Yes | No | No | No | No | No | Yes |
| Data center layout | Yes | No | No | No | No | No | No | Yes |
| New cooling techiques | No | No | No | No | Yes | No | No | Yes |
| Multi-DC | No | No | No | No | Yes | No | Yes | Yes |

Table 1: Summary of features for most common simulators in the state-of-the-art

DEVS enables the representation of a system by three sets and five functions: input set ($X$), output set ($Y$), state set ($S$), external transition function ($\delta_{\text{ext}}$), internal transition function ($\delta_{\text{int}}$), confluent function ($\delta_{\text{con}}$), output function ($\lambda$), and time advance function ($ta$).

DEVS models are of two types: atomic and coupled. Atomic models are directly expressed in the DEVS formalism specified above. Atomic DEVS processes input events based on their model's current state and condition, generates output events and transition to the next state. The coupled model is the aggregation/composition of two or more atomic and coupled models connected by explicit couplings. Particularly, an atomic model is defined by the following equation:

$$A = \langle X, Y, S, \delta_{\text{ext}}, \delta_{\text{int}}, \delta_{\text{con}}, \lambda, ta \rangle \tag{1}$$

where:

- $X$ is the set of inputs described in terms of pairs port-value: $\{p \in IPorts, v \in X_p\}$.
- $Y$ is the set of outputs, also described in terms of pairs port-value: $\{p \in OPorts, v \in Y_p\}$.
- $S$ is the set of sequential states.
- $\delta_{\text{ext}} : Q \times X^b \rightarrow S$ is the external transition function. It is automatically executed when an external event arrives to one of the input ports, changing the current state if needed.
  - $Q = (s,e)s \in S, 0 \leq e \leq ta(s)$ is the total state set, where $e$ is the time elapsed since the last transition.
  - $X^b$ is the set of bags over elements in $X$.
- $\delta_{\text{int}} : S \rightarrow S$ is the internal transition function. It is executed right after the output ($\lambda$) function and is used to change the state $S$.
- $\delta_{\text{con}} : Q \times X^b \rightarrow S$ is the confluent function, subject to $\delta_{\text{con}}(s, ta(s), \emptyset) = \delta_{\text{int}}(s)$. This transition decides the next state in cases of collision between external and internal events, i.e., an external event is received and elapsed time equals time-advance. Typically, $\delta_{\text{con}}(s, ta(s), x) = \delta_{\text{ext}}(\delta_{\text{int}}(s), 0, x)$.
- $\lambda : S \rightarrow Y^b$ is the output function. $Y^b$ is the set of bags over elements in $Y$. When the time elapsed since the last output function is equal to $ta(s)$, then $\lambda$ is automatically executed.
- $ta(s) : S \rightarrow \Re_0^+ \cup \infty$ is the time advance function.

The formal definition of a coupled model is described as:

$$M = \langle X, Y, C, EIC, EOC, IC \rangle \tag{2}$$

where:

- $X$ is the set of inputs described in terms of pairs port-value: $\{p \in IPorts, v \in X_p\}$.
- $Y$ is the set of outputs, also described in terms of pairs port-value: $\{p \in OPorts, v \in Y_p\}$.
- $C$ is a set of DEVS component models (atomic or coupled). Note that $C$ makes this definition recursive.
- $EIC$ is the external input coupling relation, from external inputs of $M$ to component inputs of $C$.
- $EOC$ is the external output coupling relation, from component outputs of $C$ to external outputs of $M$.
- $IC$ is the internal coupling relation, from component outputs of $c_i \in C$ to component outputs of $c_j \in C$, provided that $i \neq j$.

Given the recursive definition of $M$, a coupled model can itself be a part of a component in a larger coupled model system giving rise to a hierarchical DEVS model construction.

In the last decade, many DEVS M&S engines have come into existence. All of them offer a programmer-friendly Application Programming Interface (API) to define new models using a high level language. To implement our simulation models, we have used a cross-platform DEVS simulator, called xDEVS (Risco-Martín, , Mittal, Jiménez, Zapater, and Correa 2017).

## 3 SFIDE ARCHITECTURE

In this Section we show a top-down view of the data center DEVS model. From a real-world perspective, the actual structure of our model is constituted by the instantaneous workload, the measurement of the outside temperature, and a collection of *R* rooms. Each room contains a resource manager, and a set of *I In Row Coolers (IRCs)*. Each IRC is linked to *K* racks, and each rack to *S* servers. Finally, the set of *R* rooms are working in conjunction to the cooling system, which is formed by a pump, a tower and the chiller.

In the following, we describe each one of the coupled models that compose the DEVS model, as well as their components.

### 3.1 DataCenter: the root coupled model

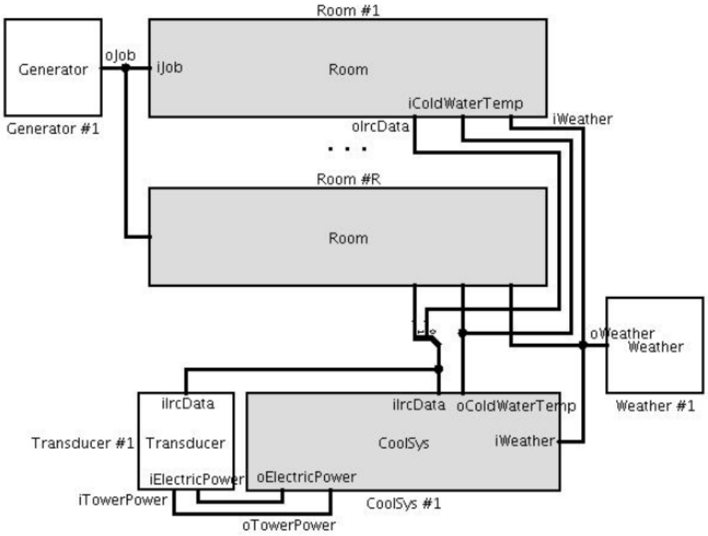


Figure 1: DataCenter: the root coupled model.

Figure 1 represents the top view of the data center model. According to the DEVS coupled model definition in equation 2, we may found the following components into this coupled model:

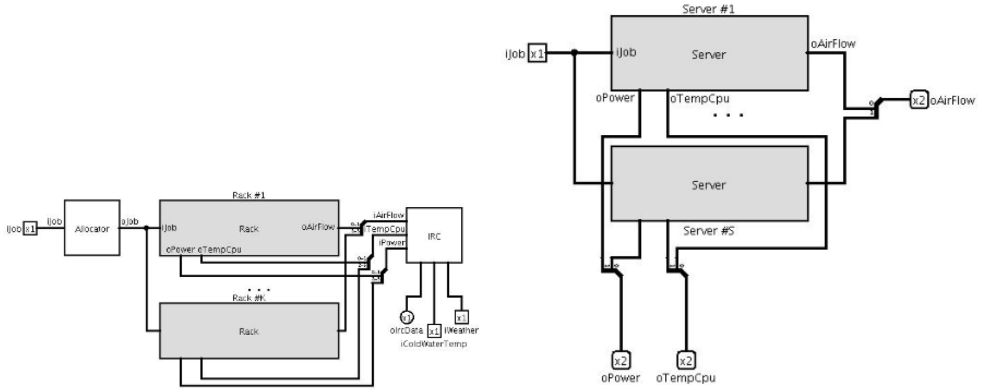
- **Generator**: The atomic component can work on two modes: on-line or off-line. In the off-line mode, it takes the set of jobs from a job logger. In the on-line mode jobs are generated by real applications and thus this atomic model acts as a wrapper.
- **Room**: There are *R* `Room` coupled models. The `Room` coupled model is described in Section 3.2.
- **Weather**: This atomic model takes the outside temperature, which is sent to other coupled or atomic models, as is depicted in Figure 1.

- **CoolSys**: This coupled model represents de cooling system. The `CoolSys` coupled model is described in Section 3.5.
- **Transducer**: This atomic model has been designed to store all the results of the simulation.

Regarding the rest of the elements in equation 2, the `DataCenter` coupled model does not contain inputs ($X = \emptyset$), outputs ($Y = \emptyset$), external input connections ($EIC = \emptyset$) or external output connections ($EOC = \emptyset$). The set of internal connections is clearly depicted in Figure 1

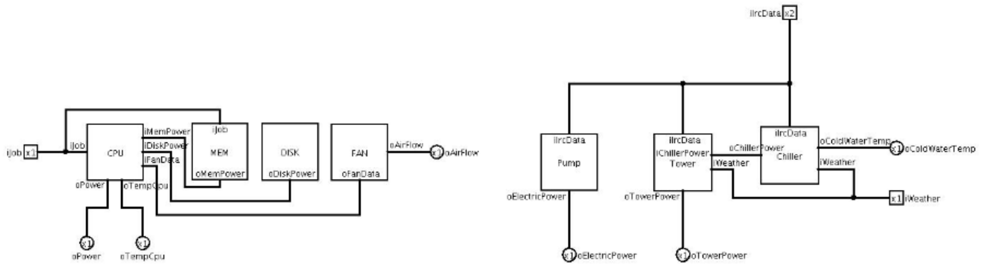
### 3.2 Room

Figure 2(a) shows the `Room` coupled model. As stated above, a `DataCenter` coupled model contains $R$ instances of this type. Following equation 2, the set of inputs are provided through the ports `iJob`, `iColdWaterTemp` and `iWeather`. These ports will contain the current job being allocated, the water temperature and the outside temperature, respectively. The set of outputs are sent through the port `oIrcData`, which basically consists of the energy consumed by all the components in the `Rack` coupled model.



(a) Room coupled model

(b) Rack coupled model



(c) Server coupled model

(d) CoolSys coupled model

Figure 2: Room, Rack, Server and CoolSys coupled models

The set of external input connections, external output connections and internal connections can be easily determined following Figure 2(a).

The set of components of the `Room` coupled model is described in the following:

- **Allocator**: The `Allocator` atomic model assigns each job received to a given rack, server and CPU. The assignment is performed trying to minimize the total energy consumption.
- **Rack**: A `Room` coupled model contains $K$ racks. Thus, a `DataCenter` coupled model has $R \times K$ `Rack` instances. This coupled model is explained in Section 3.3.
- **IRC**: The `IRC` atomic model is responsible of gathering all the information coming from servers and racks, needed by the cooling system.

### 3.3 Rack

The `Rack` coupled model does not have behavior, i.e., does not have atomic models. As depicted in Figure 2(b), this coupled model contains just other coupled models.

It has one single input port, `iJob`, to receive the current processing job, and three output ports: `oPower`, `oTempCpu` and `oAirFlow`. These ports send the total energy consumed by servers, CPU temperatures, and the air flow produced by fans, respectively. The set of external input connections, external output connections and internal connections are easy to obtain from Figure 2(b).

There is just one type of component inside the `Rack` coupled model, the `Server` coupled model. Indeed, a `Rack` coupled model includes $S$ `Server` instances. Thus, a `DataCenter` coupled model contains $R \times K \times S$ `Server` coupled models.

### 3.4 Server

Figure 2(c) shows the `Server coupled model`. This model is in charge of the calculation of energy consumption coming from processing elements and fan speed.

There is one single input port `iJob` to inject the current processing job. There are three output ports: `oPower`, `oTempCpu` and `oAirFlow` used to send the total energy consumed by CPU, memory, disks and fans, the CPU temperature, and the air flow produced by fans, respectively. As always, the set of external input connections, external output connections and internal connections are clearly shown in Figure 2(c).

The set of components of the `Server` coupled model are shown below:

- **CPU**: The `CPU` atomic model applies the corresponding power and temperature models to compute the energy consumption and the current CPU temperature. It also adds the power consumed by the other three components.
- **MEM**: This atomic model computes the energy consumed by the memory subsystem.
- **DISK**: The `DISK` atomic model computes the energy consumed by the set of disks installed in this server.
- **FAN**: This atomic model computes both the energy consumed by the fans and the air flow.

### 3.5 CoolSys

Figure 2(d) shows the coupled model implemented to simulate the Cooling System.

| DC Size | Number of Logs | Racks | Servers | Cores |
|---|---|---|---|---|
| Small | 421554 | 6 | 108 | 1728 |
| Medium | 751775 | 18 | 324 | 5184 |
| Large | 1812667 | 54 | 972 | 15552 |

Table 2: Simulation scenarios

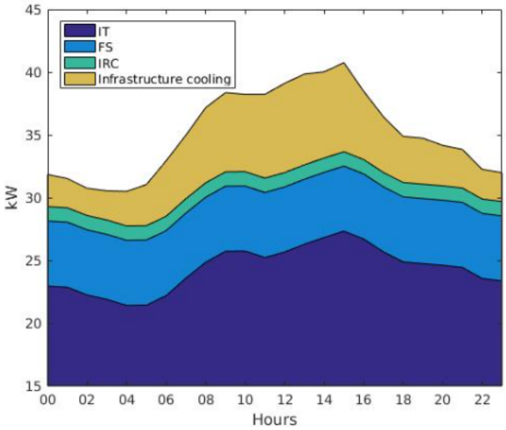


Figure 3: Small DC consumption

There are two input ports: `iIrcData`, with all the power consumed by all the rooms, and `iWeather` with the outside temperature. There are three output ports: `oColdWaterTemp` with the water temperature, and `oTowerPower` and `oElectricPower` with the power consumed by the whole cooling system.

As in previous coupled models, the set of external input connections, external output connections and internal connections are clearly shown in Figure 2(d).

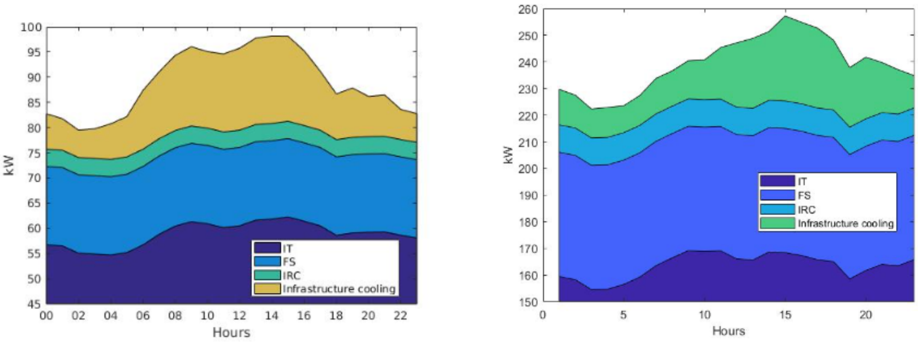
The set of components stored in `CoolSys` are described below:

- **Pump**: This is the `Pump` atomic model that computes the power consumed by the pump taking into account pump efficiency and pressure and water flow.
- **Chiller**: The `Chiller` atomic model where the power consumed by the chiller and the **Tower** are computed.

This DEVS model is able to represent a wide variety of data centers, that can be simulated in a sequential, parallel or distributed manner. As it follows the DEVS formalism, each component presented above can be replaced by an actual structure, or a library implementation correctly encapsulated in a DEVS wrapper, for instance.

## 4 CASE STUDY

In this section we analyze the behavior of the simulator comparing the same workload traces in different data center sizes. To perform a fair comparison, we have adapted the number of jobs to each data center. This is possible working on off-line mode, where all the jobs are taken from a logger file.



(a) Medium DC consumption

(b) Large DC consumption

Figure 4: Other simulations

These data (workload characterization) were collected within a time stamp from March 3rd to October 20th 2012, in CEA-Curie (HPC-Curie 2017). We have selected this HPC data center, which is located in France, Europe because the logger file provided and the size of the data center permit us running simulations without creating new traces which would fake the final results. This data center is composed by 93312 cores. As mentioned before, simulations were ran over three different scenarios we can see described in Table 4.

Figure 3 shows the simulation results for the smallest scenario. We can see how IT energy consumption is the maximum consumer of the data center. IT also sets the consumption trend. Due to the increase of the consumption of the IT (main hours of the day) we can observe how the cooling energy raises over the average. In that moments all the cooling elements are required to keep servers working.

Figure 4 depicts the simulation results for both the medium and large scenarios. It shows how FS and IRC energy scale in the same range as the DC does. Indeed, Figures 4(a) and 4(b), medium and large scenarios, respectively, show the same trends as in Figure 3.

As all these three figures depict, a higher occupation of the DC means a higher energy consumption in terms of IT and the cooling system (CoolSys in the DEVS model) while consumption of the fans and IRCs keeps constant because the refrigeration must remain constant. At the same time, however, more resources are required so the cooling system also increases its energy consumption. The cooling system includes the consumption of the tower, the pump and the chiller.

As can be seen, the DEVS representation is able to tackle complex models. Moreover, scalability is assured, since current DEVS implementations can simulate models in sequential, parallel or distributed manner without changing a single line of the DEVS model original implementation.

## 5 CONCLUSIONS AND FUTURE WORK

Data centers are huge power consumers and have very high operational costs. The study the impact of variables very different in nature such as layout workload allocation can be only performed through simulation. Current simulation approaches are focused only on data room thermal dynamics or target only a particular phase of data center operations, like workload allocation.

In this paper we have introduced a data center model. To this end, we have used the well-known DEVS formalism. DEVS is a modular and hierarchical modeling formalism, with all of the advantages and uses of simulation systems, such as: completeness, verifiability, extensibility, and maintainability and allows execution of Monte Carlo simulations, parallel simulation using threads or distributed using web services, as an example. In this paper we have used the xDEVS open source C++ library.

The DEVS model computes energy consumed by each component of the data center. This model has been validated using three synthetic scenarios of three different sizes: small, medium and large. This work allows to test the system's behavior under real conditions of job allocation. This simulator has been developed departing from (Zapater, Turk, Moya, Ayala, and Conkun 2014) simulator which results, as mentioned on section III.b were validated with real traces. Experiments were run in both simulators obtaining same results for each scenario.

A future work we propose the implementation of hybrid models combining DEVS, third-party libraries (like iNET) and real hardware devices.

## ACKNOWLEDGMENTS

The authors would like to thank Mr. Fernando Baraibar López and Mr. Roberto Montero Cobo de Guzmán, who implemented the initial structure of the framework presented in this paper as part of their senior project. This work is supported by the Spanish Ministry of Economy and Competitiveness under research grant TIN 2015-65277-R.

## REFERENCES

Calheiros, R. N., R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya. 2011, January. "CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms". *Softw. Pract. Exper.* vol. 41 (1), pp. 23–50.

DOE ASCAC Subcommittee 2014, february. "Top Ten Exascale Research Challenges". Technical report, U.S. Department of Energy (DOE).

Engbers, Norbert and Taen, Eric 2014, November. "Green Data Net. Report to IT Room INFRA". European Commision. FP7 ICT 2013.6.2.

HPC-Curie 2017. "Curie High Performance Computing". http://www-hpc.cea.fr/index-en.htm.

Google Inc. 2017. "Efficiency: How we do it".

Alejandro Lucero 2011. "Simulation of batch scheduling using real production-ready software tools".

Malik, A., K. Bilal, S. Malik, Z. Anwar, K. Aziz, D. Kliazovich, N. Ghani, S. Khan, and R. Buyya. 2015. "CloudNetSim++: A GUI Based Framework for Modeling and Simulation of Data Centers in OMNeT++". *Services Computing, IEEE Transactions on* vol. PP (99), pp. 1–1.

Matt Stansberry, J. K. 2013. "Uptime Institute 2013 Data Center Industry Survey". Technical report, Uptime Institute.

Núñez, A., J. L. Vázquez-Poletti, A. C. Caminero, G. G. Castañé, J. Carretero, and I. M. Llorente. 2012, March. "iCanCloud: A Flexible and Scalable Cloud Infrastructure Simulator". *J. Grid Comput.* vol. 10 (1), pp. 185–209.

Risco-Martín, J. L., , S. Mittal, J. C. F. Jiménez, M. Zapater, and R. H. Correa. 2017. "Reconsidering performance of DEVS modeling and simulation environments using the DEVStone benchmark". *SIMULATION Transactions of The Society for Modeling and Simulation International* vol. 1 (1), pp. 1000–1017.

Rodrigues, A., K. Bergman, D. Bunde, E. Cooper-Balis, K. Ferreira, K. Hemmert, B. Barrett, C. Versaggi, R. Hendry, B. Jacob, H. Kim, V. Leung, M. Levenhagen, M. Rasquinha, R. Riesen, P. Rosenfeld, M. Del

Carmen Ruiz Varela, and S. Yalamanchili. 2012. “Improvements to the structural simulation toolkit”. In *SIMUTools 2012 - 5th International Conference on Simulation Tools and Techniques*, pp. 190–195.

Singh, U., A. K. Singh, S. Parvez, and A. Sivasubramaniam. 2010. “CFD-based Operational Thermal Efficiency Improvement of a Production Data Center”. In *Proceedings of the First USENIX Conference on Sustainable Information Technology*.

CoolSim Software 2016. “CoolSim 4.4”.

Varga, A. 2001. “The omnet++ discrete event simulation system”. In *In ESM’01*.

Yeo, S., and H.-H. S. Lee. 2012. “SimWare: A Holistic Warehouse-Scale Computer Simulator”. *Computer* vol. 45 (9), pp. 48–55.

Zapater, M., A. Turk, J. M. Moya, J. L. Ayala, and A. K. Conkun. 2014. “Dynamic Workload and Cooling Management in High-Efficiency Data Centers”. In *Proceedings INTERNATIONAL GREEN and SUSTAINABLE COMPUTING CONFERENCE*.

Zeigler, B. P., H. Praehofer, and T. G. Kim. 2000. *Theory of Modeling and Simulation. Integrating Discrete Event and Continuous Complex Dynamic Systems*. 2 ed. Academic Press.

## AUTHOR BIOGRAPHIES

**IGNACIO PENAS** received a degree in telecommunication engineering from the Technical University of Madrid. He is currently working on GreenLSI researching group at Technical University of Madrid. His research interests include energy-efficient processing in data centers and machine learning modeling. His email address is nacho@die.upm.es.

**MARINA ZAPATER** received a degree in telecommunication engineering and a degree in electronic engineering from the Technical University of Catalunya, Spain. She also received her Ph.D. from the Technical University of Madrid, Spain. She is currently an Assistant Professor in the Department of Computer Architecture and Automation at Complutense University of Madrid. Her research focuses on thermal optimization of complex heterogeneous systems, proactive and reactive thermal-aware optimization of data centers. Her email address is marinzap@ucm.es.

**JOSÉ L. RISCO-MARTÍN** received his Ph.D. from Complutense University of Madrid, and currently is Associate Professor in the Department of Computer Architecture and Automation at Complutense University of Madrid. His research interests include computer aided design, optimization and discrete event simulation. His email address is jlrisco@ucm.es.

**JOSÉ L. AYALA** received the M.Sc. and Ph.D. degrees in telecommunication engineering from the Technical University of Madrid, Madrid, Spain, in 2001 and 2005, respectively. He is currently an Associate Professor in the Department of Computer Architecture and Automation, Complutense University of Madrid, Madrid, Spain. His email address is jayala@ucm.es.