2014 28th International Conference on Advanced Information Networking and Applications Workshops

Towards Simulating the Internet of Things

[1]Stelios Sotiriadis, [2]Nik Bessis, [2]Eleana Asimakopoulou and [3]Navonil Mustafee
[1]Intelligent Systems Laboratory, Technical University of Crete, Greece
[2]Distributed and Intelligent Systems Research Centre, University of Derby, United Kingdom
[3]Centre for Innovation and Service Research, University of Exeter, United Kingdom
[1]s.sotiriadis@intelligence.tuc.gr [2]n.bessis@derby.ac.uk [2]eleana.asimakopoulou@googlemail.com [3]N.Mustafee@exeter.ac.uk

Abstract — **There is an increasing interest in Internet of Things (IoT) and healthcare is considered to be one of the most common applications of it. Using the IoT paradigm, various devices including smart-phones and sensor-embedded healthcare applications can be used for monitoring health. In this study, we model an IoT use case scenario with regard to monitoring the activities associated with health. In particular, we present our use case using the SimIoT extended simulation toolkit to demonstrate the various functions and the interactions occurring within the IoT-enabled healthcare context. Specifically, we extend the functionalities of the SimIC simulation toolkit by adding the IoT layer that incorporates IoT devices which generated data for the private clouds. We focus our experimental analysis from the perspective of cloud performance to illustrate the turnaround and makespan of the system.**

Keywords: Internet of Things, Simulation toolkits, Job processing, Heathcare, Emergency scenarios

I. INTRODUCTION

Simulators and analysis tools for distributed systems have increasingly been central to systems' development. It provides the stakeholders an opportunity to test the environment in terms of its configuration and resource deployment prior to changes being made to production systems. . Further, real test bed experiments are difficult to be performed due to the large number of different requirements and administrative costs. In the context of designing IoT-centric/IoT-based distributed systems, the necessity for simulation is further necessitated by the fact that IoT paradigm encompasses integration of various sensors, that in most cases are expensive to be used only for testing. So, the simulation toolkits allow developers to model and test their research hypothesis prior to the actual hardware configuration [1]. This includes the need to evaluate various resource management phases (users and resources) through several scenarios where the actual experiments are limited to the real test bed system's scale and capabilities.

In this work we focus on the IoT paradigm [17] and we aim to implement and test the actual behavior of such system. We anticipate that the back-end operations are executed in a cloud environment. In the context of clouds and IoT, the existing systems do not permit extended testing based on specific resource management objectives [2]. Thus we identify the simulators that could meet dynamic information processing where users are IoT devices that schedule request for services (jobs) in private clouds. It is an essential requirement that we consider a highly heterogeneous and dynamic environment.

To demonstrate our solution, we present a use case of an IoT health monitoring system in order to manage emergency situations. This is based on the utilisation of short range and wireless communication devices that emerge opportunities in enhancing the management of emergency situations. Through this, smart phones and sensor-embedded mobile healthcare devices achieve remote communication through Wi-Fi, 3G or GPS networks [18].

Based on this, we first present an analysis of large-scale simulation systems in order to identify their applicability to IoT scenarios (Section II). In Section III we present the SimIoT toolkit that extends SimIC capabilities in terms of user submissions. In particular, users are sensors that submit requests to a cloud environment for information processing. Following this, in Section IV we highlight the healthcare use case that utilises smart-phones to weave an IoT-based distributed system. Section V discusses the simulation study. Section VI is the concluding section of the paper; it presents a discussion on future work phases and summarises the research presented in this paper.

II. THE ANALYSIS OF SIMULATION TOOLKITS

This section presents an overview of the simulation toolkits that have been proposed for large-scale resource management. As the scope of the study is restricted to IoT environments that act as the source of input data for the clouds, our focus is primarily on job processing and not on network simulation toolkits. The readers should take note that a detailed analysis of resource management approaches is presented in [16]. A crucial requirement for IoT scenarios is the interconnection in a complex and real-time topology [17] of IoT devices requests during simulation.

MicroGrid [3] is an environment that offers the basic tools for performing simulation experiments in the grid environment. Authors claim that MicroGrid is a feasible experimental toolkit for scheduling systems that are not real-time. GridSim [4] is a toolkit for the simulation of schedulers and brokers in the grid environment. Like MicroGrid, the default version of GridSim does not support experimentation pertain to real-time scheduling; further, virtualisation is not implemented. In contrast, GangSim [5] includes the notion of virtual organisations and multi-sites. It enables repeatable and controllable experimentation with dynamic resource

978-1-4799-2652-7/14 $31.00 © 2014 IEEE
DOI 10.1109/WAINA.2014.74

management techniques. One critique of GangSim is that it does not consider virtualisation and heterogeneity (diversity of requirements) [1].

GSSIM [6] is a simulation environment that is based on GridSim and is used primarily for experimenting with scheduling strategies pertaining to multi-level and heterogeneous grid systems. It offers a flexible alternative for improving the simulation execution speed associated with traditional GridSim simulator; however, it does not allow the representation of certain optimal schemes and real-time scheduling. Alvio [7] is a simulation model based on C++ that evaluates traditional HPC scheduling approaches. It offers a prediction module for incorporating past performance requirements which can be used to inform future job scheduling decisions. However, some authors (e.g., [8]) are critical -of the heterogeneity aspects of the system. DGSim [8] offers a framework for developing simulation schedulers of various grid resource management architectures. Authors claim that DGSim considers inter-operation of grids and relevant dynamics. However, an implementation of the simulator has not yet been distributed to the public.

SimGrid [9] offers the core functionalities for simulating distributed applications in large-scale heterogeneous distributed systems. It allows testing on non-centralised and heterogeneous schedulers that aim at dynamic resource availability models without the support of virtualisation. Alea 2 [10] is a simulation framework which supports heterogeneous resources and dynamic job flows; however it does not presently include support for virtualisation. MONARC [11] is a tool for simulating frameworks aimed at optimising resource allocation in a distributed computing environment. This toolkit allows the simulation of data replication and scheduling.

SmartGRID [12] offers a decoupled and layered structure and an interoperable infrastructure for grid resources by utilising fully decentralised and bio-inspired algorithms. For supporting scheduling decisions, SmartGRID integrates a simulator that provides services as group communication through asynchronous message passing and resource discovery. Authors claim that SmartGRID supports heterogeneity.

CloudSim [1] overcomes the absence of virtualisation technology. CloudSim offers a seamless model with virtual machine (VM) support. This could allow the testing of more advanced solutions such as process migration at runs time. The aforementioned simulation environments have been developed in order to mimic advanced resource management decisions of different systems e.g. HPC, grids and clouds. It has been proposed by [2] that the conventional schemes, e.g. GangSim or GridSim or their alternatives such as GSSim, are functionality limited to address directly the cloud processing requirements. In order to address these shortcomings and to further research in the area of network simulation of the different topological variations of the Cloud environment, the *'Simulating the Inter-Cloud' (SimIC)* environment [13] has been proposed. SimIC aims to model an inter-cloud facility wherein multiple clouds collaborate with each other for distributing service requests in regard to the desired simulation setup. It allows, (a) flexible user submissions (e.g. in the form of sensors), (b) the development of the designs pertaining to the various topologies and entities for IoT scenarios, (c) supports simulation of heterogeneous environments and (d) includes support for virtualisation, and (e) supports distributed computing environments that are subject to real-time constraints. The package encompasses the fundamental entities of the inter-cloud meta-scheduling framework [14] such as users, meta-brokers, local-brokers, datacenters, hosts, hypervisors and virtual machines (VMs).

Table 1 presents a summary of the network simulators and their essential characteristics (VM denotes the virtualisation management and the Msg as the messaging mechanism). It shows that SimIoT could expand SimIC functionality in terms of input from IoT devices. The essential features include the VM management, the messaging based on real-time constraints (e.g. certain system responses) and IoT inputs.

Table 1: Cloud configuration parameters for input in SimIoT

Simulator/ Charact.	Hetero-geneity	Inter-operability	VM	Re-scheduling	Msg	IoT
Alea	X					
GangSim	X	X				X
SimGrid	X	X				X
GridSim	X	X				X
GSSim	X	X				X
SmartGrid	X	X		X		X
MONARC	X	X	X		X	X
CloudSim	X	X	X			
SimIC	X	X	X	X	X	
SimIoT			X		X	X

III. THE ARCHITECTURE OF THE SIMIOT TOOLKIT

The SimIoT is a hybrid implementation of the SimIC in terms of input users. As mentioned earlier, data is generated from devices like sensors that automatically submit information processing requests to the cloud. The toolkit therefore includes the design of entities that communicate with each other by sending events that represent messages. Every message can potentially consist of a number of information items that can be utilised by the inter-connected entities. An analysis of the message exchanging optimisation (MEO) model is presented in [15]. The approach includes a set of algorithms for effective message exchanging in distributed systems. Figure 1 illustrates the activity diagram with the starting and the ending points for the communication of two entities in terms of messaging and real-time information retrieval (e.g. current latency).

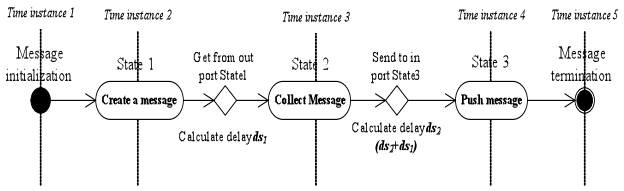

Figure 1: The activity diagram of the SimIoT events flows and time instances

The message initialisation takes place at time instance 1, wherein a message is created at *State 1*. Subsequently, *State 2* collects the message (retrieved from out port *State 1*) and sends it to port *State 3*. The simulation time instance progresses from time 2 to time 3 and then to time instance 4. Finally, the message is terminated (or initialised) from another state in order to continue the information exchanging. In Figure 1, all the three states are referring to one entity alone (e.g. the broker) that creates, collects and pushes the message to the next entity.

Figure 2 shows the high-level SimIOT architecture. It includes the user site where the IoT device input system retrieves information from the sensors. It then forwards the requests to the communication broker that is responsible for translating the contextualised information (e.g. value of 30 is a temperature property). Subsequent to this the broker generates a request for information processing and forwards this request to the default SimIC cloud entities. These are the broker, for example,meta-scheduler for management inter-operations, the datacenter that includes the physical hosts, the VMs and the storage, the hypervisor along with its policies for scheduling, messaging and virtualisation (as described in [13]). We correlate each message to a cloud service request submission that sends a data processing request to a VM for execution.

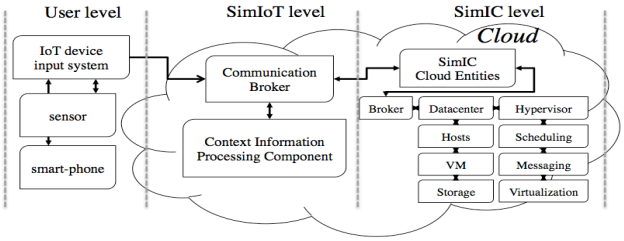

Figure 2: The high-level architecture of the SimIoT

The SimIoT architecture encompasses a three-layered structure which it is illustrated in Figure 3.

- In layer 1 the entities representing the objects of the system are included. Each class incorporates this design in order to define the actual behavior (layer 2) of entities that are the cloud resources. Specifically, the core classes are IoT device, Meta-broker, Local-broker, Datacenter, Hypervisor, Hosts, VMs and Bucket. In brief, initialisation takes place when the user initialises communication with the meta-broker that represents the IoT content. The latter acts on behalf of the user in order to forward the request to low-level resources (local or remote sites). The broker that monitors service life cycle executes this procedure. In addition, a context component is responsible for translating the contextualised information. The datacenter represents the low level infrastructure and it is the place wherein requests are forwarded to hypervisor for VM deployment.
- Layer 2 incorporates the behavior of the SimIC that represents the actions happening within the simulator. The core features are the utilisation of ports, functionalities and constraints that demonstrate the actual behavior of the entity. Each class contains at least one port for either input or output of messages and is linked to the other entities. In addition, it incorporates mechanisms for collecting messages, taking decisions (based on policies) and forwarding to the entity decided for delegation. The communication is based on tags that are assigned to messages during exchange and are the means of identifying the origin of the message and the required operation of the responder. The IoT tag is a class of tags to denote the data submissions pertaining to the unique IoT devices that constantly feed information to the cloud environment. The event tag is a set of low-level infrastructure tags that are generated dynamically by the system for operational reasons.

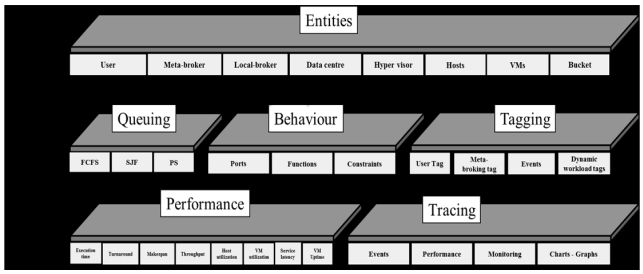

Figure 3: The SimIC layered structure

- Layer 3 includes the performance and tracing operations. The performance measures include execution time of the VM, service turnaround time, makespan of the service, service throughput, host utilisation levels, VM utilisation levels and service latencies. The formulas for the aforementioned metrics are presented in [14]. These metrics are implemented in the default version of the SimIC. Finally, the tracing includes the logging of events and their inter-exchanges, performance results, monitoring of the whole service submission and production of charts and graphs with regard to the simulation scenario. Next, we present the use case.

IV. USE CASE: HEALTH-MONITORING SYSTEM FOR EMERGENCY SITUATIONS

The utilisation of short range and wireless communication devices present several opportunities in monitoring the health of patients, especially in emergency situations. Those using smart phones and similar devices could establish remote communication through Wi-Fi, 3G or bluetooth networks. Our basic scenario involves users that utilise their personal devices for communication within an interactive environment. Users have a mobile device/smart phone which enables them to communicate through local Wi-Fi networks and access the Internet services. Thus, the users have access to the local services offered by the environment in question (in our case this is healthcare) along with their personalised profiles and services that are available through the Internet.

The emergency scenario defined in our use case begins when a user is in a critical situation and is in need of a health care provision. As the users have mobile devices/smart phones they can start the process of exploration of their health status. This is achieved when the users retrieve their personalised health profile. Subsequently, the users interact with the environment through their smart device by submitting requests to the system through a sensor based notification service. We assume that the sensors collect information about the atmosphere and data that is manually added by the users (using a pre-installed application).

The information enables the smart environment to take intelligent decisions based on users' personal health profiles and dynamic capabilities of sensors. For example, let us assume that a patient is in an emergency situation and the monitoring system automatically collects information and forwards into the appropriate administrative division for evaluation. Then the system makes suggestions for specific actions by gathering information from the personal profile that is stored and available from one or more of the users' registered devices. It should be mentioned that the profile being considered in this case study is based on recent user experiences. A business intelligence process shall be used to decide the actions to be pushed in the users mobile device based on various constrains.

Another use case example is the airport terminal disaster management case. We assume there are users that are moving in an airport terminal. The users are monitored either by a) accessing their smart devices, b) accessing static beacons (e.g. CCTV to count people) or c) dynamic beacons (e.g. information coming from smart phones of staff). The users are moving randomly however with a target direction to a key point location during time (e.g. the departure gates, or baggage claim etc.). In this case the users are moving randomly with a target direction to the exit points. We collect their position and their health status and we make suggestions for rescuers. The assumption is that the airport has a private cloud that offers data processing capabilities, thus no data are available outside this area.

Each time a user is connected to the airport private cloud they submit a request for job execution. This includes any contextualised information about the patient (e.g. the temperature). The communication broker identifies the property and forwards it to the appropriate entity for evaluation. We assume that each request is executed into a VM that has been configured in previous steps. Thus, each request is correlated to a job submission because it requires a certain computational power for being executed. The assumption is that that there is a data processing system pre-installed into the VM. The next section demonstrates a simple SimIoT simulation where patients send data to the private cloud for information processing.

V. BUILDING A SIMIOT SIMULATION

This case scenario involves the experimental input of 160 identical jobs submitted by 16 IoT devices (10 jobs per user) and the cloud has available resources for job execution. Each request follows the following process: IoT devices request for computational performance and its execution is sandboxed in a VM. The performance is quantified by executing a set of programs. The execution time of the VM is calculated by (1):

$$\text{ExecTimeVM}=\text{Instruction per job}\times \text{cycles per instruction}\times \text{seconds per cycle} \quad (1)$$

So we can now calculate the execution time of an IoT device request. Based on the above example it is $100*106\text{ns.}\times 3\times 11000\times 11=3\times 10\ 5\text{ns.}=0.3\text{ ms.}$

The performance measures are given by the MIPS formula (2) that calculates the millions of instructions per second (MIPS) as a rate for operations per unit.

$$\text{MIPS} = \text{clock rate}\text{CPI}*10-6 \quad (2)$$

The execution includes the configuration of tables 2 and 3 (BW defines the bandwidth).

Table 2: Cloud configuration parameters for input in SimIoT

VM features	MIPS	RAM	HD	BW	Host Num	Host 1 Cores	Host 2 Cores
Values	1000	2048	10^5	10000	2	4	2

Table 3: IoT device configuration parameters for input in SimIoT

VM Requirements	CPU size	RAM	MIPS	Bandwidth	Cores
Experiment Values	1000	512	1000	1000	1

In this case a dynamic workload management defines the dissemination functionality. This involves that if the cloud cannot execute the job due to limited resources then it sends the job back to a queue for execution. Figure 4 shows the turnaround and the trend line of the turnaround times of SimIoT. It is apparent that the turnaround trend line shows an increasing trend for 50 to 100 job submissions; however for 100 to 160 the line shows a decreasing rate. We consider this as an improvement because the system tends to offer better performance for peak workloads.

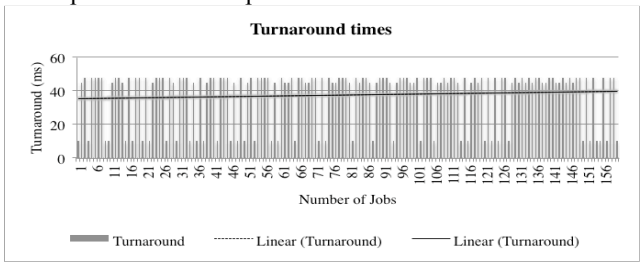

Figure 4: The turnaround and the trend line of the turnaround of SimIoT for 160 identical jobs submitted by 16 users (10 jobs per user)

Figure 5 presents the makespan values of SimIoT when 160 identical jobs submitted by 16 users in an identical case as previously. It shows that the makespan increases as more IoT devices submit requests to the system. However, we consider that the increasing rate is small as for the 1st user the makespan is around 400ms while for the 160th user the value is almost doubled (around 830ms).

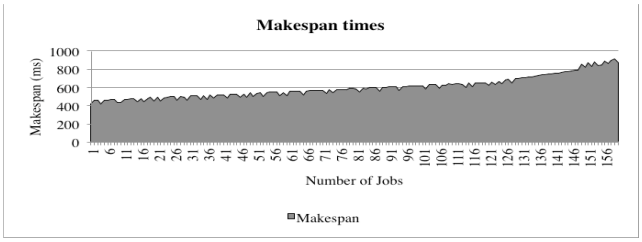

Figure 5: The makespan values of SimIoT for 160 identical jobs submitted by 16 users (10 jobs per user)

VI. Conclusions and Future Work

In this paper we have presented SimIoT, a toolkit to achieve experimentation on dynamic and real-time multi-user submissions within an IoT scenario. The toolkit is based on the SimIC that allows modelers to configure a diversity of clouds in terms of datacenter hosts and software policies wherein desired number of users could send single or multiple requests for computational power, software resources and duration of VM virtualisation.

Future research steps include the conceptualisation and the implementation of extended experiments in order to test topologies and heterogeneous IoT devices. Another aspect that is related with the IoT paradigm is to empower simulator to simulate scenarios that implement a collection of sensors that utilise the backbone of the inter-cloud infrastructure. A novel challenge is the exploration of different ranking techniques (based on job performance measures) to achieve optimisation of metrics. We also aim to implement distributed processing capabilities and explore the performance of SimIoT.

Another challenge will be to import energy efficiency measures for optimising message distribution among entities and allow effective management of IoT resources. This will increase the effectiveness of current optimisation schemes as well as will offer modularity for assisting modelers to define new entities by improving class design. Also a challenge is to add VM migration cases and design new scenarios e.g. disaster scenario backup in order to extend the applicability of the toolkit.

References

[1] Calheiros, R., N., Ranjan, R., Beloglazov, A., De Rose, A.F.C., and Buyya, R. (2011) CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms, Software: Practice and Experience (SPE), 41(1), ISSN: 0038-0644, Wiley Press, New York, USA, January, 2011, pp. 23-50

[2] Buyya, R., Ranjan, R., and Calheiros, R. N., (2010) InterCloud: Utility-Oriented Federation of Cloud Computing Environments for Scaling of Application Services, Algorithms and Architectures for Parallel Processing (2010), Volume: 6081/2010, Issue: LNCS 6081, Publisher: Springer, pp. 13-31

[3] Song, H.J., Liu, X., Jakobsen, D., Bhagwan, R., Zhang, X., Taura, K., and Chien. A. (2000) The microgrid: a scientific tool for modeling computational grids. ACM/IEEE 2000 Conference Supercomputing, pp. 53-53

[4] Buyya, R., and Murshed, M. (2002) GridSim: a toolkit for the modeling and simulation of distributed resource management and scheduling for Grid computing. Concurrency and Computation: Practice and Experience, 14(13-15), pp. 1175-1220

[5] Dumitrescu, C., Wilde, M., and Foster, I. (2004) Usage Policy at the Site Level in Grid3, GriPhyN Technical Report, 2004-71

[6] Kurowski, K., Nabrzyski, J., Oleksiak, A., and Weglarz, J. (2007) Grid scheduling simulations with GSSIM. 3rd Workshop on Scheduling and Resource Management for Parallel and Distributed Systems, Proceedings of the 13th International Conference on Parallel and Distributed Systems, Hsinchu, Taiwan, 2007

[7] Alvio Simulator (2002), Accessed August 2 2013, http://www.guim.net/fguim/alvio/index.htm.

[8] Iosup, A., Sonmez, O., and Epema, D. (2008) DGSim: Comparing grid resource management architectures through trace-based simulation, In Euro-Par 2008-Parallel Processing, Springer, 2008, pp. 13-25

[9] Casanova, H., Legrand, A., and Quinson, M. (2008) SimGrid: A generic framework for largescale distributed experiments. *Tenth International Conference on Computer Modeling and Simulation*, IEEE, 2008, pp. 126-131

[10] Klusacek, D., and Rudova, H. (2010) Alea 2: job scheduling simulator, *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques*, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2010, pp. 1-10

[11] Dobre, C., and Stratan, C. (2004) MONARC Simulation Framework, Transactions on Automatic Control and Computer Science, 49(63), 2004, ISSN 1224-600X Special issue on 3rd Edition of RoEduNet International Conference, Timisoara, Romania

[12] Huang, Y., Bessis, N., Sotiriadis, S., Brocco, A., Courant, M., Kuonen, P. and Hirsbrunner, B. (2009) "Towards an Integrated Vision across Inter-Cooperative Grid Virtual Organizations in Lee", Y., Kim, T., Fang, W. and Slezak, D. (eds). Future Generation Information Technology, Lecture Notes in Computer Science, Springer, ISBN: 3-642-10508-4, p.p.: 120-128

[13] Sotiriadis, S., Bessis, N., Antonopoulous, A. and Anjum, A. (2013) SimIC: Designing a new Inter-Cloud Simulation Platform for Integrating Large-scale Resource Management, 27th IEEE International Conference on Advanced Information Networking and Applications (AINA-2013), March 25-28, Barcelona, pp. 90-97

[14] Sotiriadis, S., Bessis, N., Kuonen, P. and Antonopoulous, A. (2013) The Inter-cloud Meta-scheduling (ICMS) Framework, 27th IEEE International Conference on Advanced Information Networking and Applications (AINA-2013), March 25-28, Barcelona, pp 64-73

[15] Bessis, N., Sotiriadis, S., Pop, F. and Cristea, V. (2013) Using a Novel Message-Exchanging Optimization (MEO) Model to Reduce Energy Consumption in Distributed Systems, Simulation Modeling Practice and Theory, Elsevier, ISSN: 1569-190X DOI http://dx.doi.org/10.1016/j.simpat.2013.02.003

[16] Sotiriadis, S., Bessis, N. and Antonopoulos, N. (2011) Towards inter-Cloud Schedulers: A Survey of meta-Scheduling Approaches, 6th IEEE International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC-2011), Barcelona, Spain, October 26-30, 2011, ISBN: 978-0-7695-4531-8, pp. 59-66

[17] Zelenkauskaite, A., Bessis, N., Sotiriadis, S. and Asimakopoulou, E. (2012) Interconnectedness of Complex systems of Internet of Things through Social Network Analysis for Disaster Management, 3rd International Workshop on Computational Intelligence for Disaster Management (CIDM-2012) in conjunction with the 4th IEEE International Conference on Intelligent Networking and Collaborative Systems (INCoS-2012), September 19-21 2012, Bucharest, Romania, ISBN: 978-0-7695-4808-1, pp. 503-508

[18] Bessis, N., Asimakopoulou, E., French, T., Norrington, P. and Xhafa, F. (2010) The Big Picture, from Grids and Clouds to Crowds: A Data Collective Computational Intelligence Case Proposal for Managing Disasters, 5th IEEE 3PGCIC-2010, Nov 4-6, Fukuoka ISBN: 978-0-7695-4237-9 pp: 351-356