Computing and Informatics, Vol. 36, 2017, 541–565, doi: 10.4149/cai_2017_3_541

HOST-BASED VIRTUAL NETWORKS MANAGEMENT IN CLOUD DATACENTERS

Dimitris Kontoudis, Panayotis Fouliras

Department of Applied Informatics
University of Macedonia
156 Egnatia str.
GR-54006 Thessaloniki, Greece
e-mail: {kontoudis, pfoul}@uom.gr

Abstract. Infrastructure management is of key importance in a wide array of computer and network environments. The use of virtualization in cloud datacenters has driven the communications and computing convergence to a common operational entity. Failure to effectively manage the involved infrastructure results as impediments in provisioning a successful service. Information models facilitate the infrastructure management and current solutions can be effectively applied in most datacenter scenarios, apart from cases where the networking architecture relies heavily on systems virtualization. In this paper we propose an information model for managing virtual network architectures, where hypervisors and computing server resources are deployed as the basis of the networking layer. We provide a successful proof of concept by managing a virtual machine-based network infrastructure acting as an IP routing platform using statistical methods. Our proposal enables a dynamic reconfiguration of allocated infrastructure resources adapting, in real-time, to variations in the imposed workload.

Keywords: Network management, hypervisor, modeling, statistical process control, cloud, datacenter

Mathematics Subject Classification 2010: 90B18, 68M11, 62P30

1 INTRODUCTION

In recent years communication networks have shown an ever increasing use of computing servers and hypervisors as active network elements. This fact, along with the virtualization concept, which has also been adopted in the computer networks field, introduces new challenges when managing such networks. Until recently, network designers had to consider physical infrastructure elements and their characteristics (e.g. routers, communication connections and bandwidth). This has changed with the introduction of computing servers in the networking architecture [7, 50]. Communication networks implementations, as in network testbed projects, network simulation environments [10] and scenario-based infrastructure management [26], increasingly use computing servers as active network nodes, as these servers allow for flexible experimentation on new architectures, protocols and services. In the Cloud Computing paradigm [3, 6] the server loads (that could be active network elements) dynamically shift between physical servers in the same or even in data centers at different geographical locations. This has been made possible due to advances in server virtualization (also referenced to as system or machine virtualization) and the introduction of the hypervisor (also known as the virtual machine monitor), the latter implemented by a specific thin software layer. Consequently, it is now possible to logically divide a physical server into several virtual ones based on the available physical resources and their characteristics. The core networking support in server virtualization and hypervisor environments is based on the IEEE 802.1Q Virtual Lan (VLAN) implementation. The hypervisor works as a virtual Ethernet switch and supports queues for each virtual LAN in the server memory [4, 33]. In this way it is possible to establish network communication across different virtual servers, implementing virtual Ethernet adapters without routing network traffic outside the physical system which both hosts them and performs the virtualization (Figure 1). The use of this mechanism provides increased security and much more flexible network deployment compared to physical network devices. These technologies are now offered by all server virtualization products available in the market. Consequently, the network *last-hop switch* has been shifted from being a pure active network element to a characteristic of the hypervisor or the physical server hardware [14]. Overall, a paradigm change is in progress in networking by which the association *user to IP address, to active network element, to physical location* may no longer apply.

New issues are, therefore, raised and need to be considered in network design, operation, monitoring and administration, namely, the involved physical server hardware resources (capacity of processors, memory, virtual switch, etc.) and the optimization of these resources in terms of performance and behavior. Managing such network architectures presents an increasing need for the infrastructure detailed characteristics to be represented in a formal, standardized and structured manner. This is facilitated by the use of suitable information models that describe the involved infrastructure components and organize their characteristics and interrelations, efficiently. Standardization enables development of suitable management

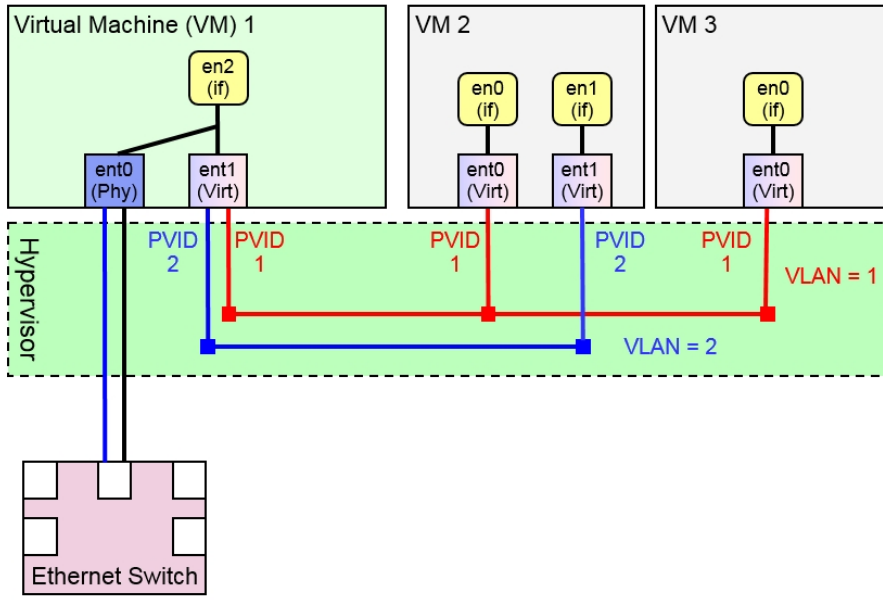

Figure 1. High-level architecture of systems networking based on the hypervisor IEEE 802.1Q virtual switch

software, formulation of accurate service level agreements (SLA) and creation of supporting environments in order to enable collateral actions, such as accounting, auditing and standards compliance. The latter need is becoming evident in several implementations due to Government and other organizations requirements (e.g. Sarbanis-Oxley, FISMA and HIPAA standards, European Union 95/46/EC directive, PCI-DSS standard pertaining to credit card computer and network systems, etc.)

At present there are very few proposals addressing the problem and details of the introduction of computing servers in the network architecture and the hypervisor as a manageable entity, in particular. The architectural, technological and operational complexity in infrastructures, where hypervisors are deployed as a core component of network virtualization, has largely been overlooked in relevant research. Moreover, existing solutions do not treat the hypervisor as a whole entity. Instead, they only reference its involvement indirectly, via abstractions of hosted virtual machine operations. This causes severe impediments in managing modern datacenter virtual networks, since there is no quality assurance of the offered service on an end-to-end basis.

Our research has been motivated by the aforementioned issues and we worked towards designing an extensible information model, able to abstract hypervisors, their components and characteristics in a variety of architectures. In this paper we

propose the *Kernel Footprint* (KF) information model based on the CIM standard (Common Information Model) [17]. The proposed model fits well within the context of highly virtualized, hypervisor-based, virtual networks present in modern Cloud datacenters. Our approach can conceptually describe physical or logical infrastructure elements participating in such infrastructures. It is, therefore, possible to facilitate efficient element management, be that a networking, computing system or other resource. Furthermore, the management application does not need to be aware of the specifics of the underlying virtualization platform. We provide a thorough proof-of-concept of our approach, by applying the model on a hypervisor-based virtual network architecture, based on the IBM Advanced Interactive Executive (AIX) operating system servers, implementing a virtual routing platform. The model is extended in order to provide support for statistical processing of infrastructure utilization data and a controller is deployed allowing dynamic resource management of the hypervisor-provisioned resources. Test results indicate that the model has actual practical applicability and can be successfully deployed in similar datacenter architectures.

The remainder of this paper is organized as follows. Section 2 provides an overview of related research. Section 3 describes the proposed model, outlines its relevance to other proposals and presents a test case implementation. A discussion of our work is provided in Section 4. We conclude in Section 5 by summarizing the findings of our research and presenting future directions.

2 RELATED WORK

Network virtualization research spans a wide variety of topics, ranging from very specific technical issues (interfacing, signaling and bootstrapping, resource and topology discovery, resource allocation, admission control, virtual nodes and virtual links, naming and addressing) to broader interest areas such as mobility management, monitoring, configuration and failure handling, interoperability issues, security and privacy. A concise survey of network virtualization research is provided in [11, 12]. Active [46], programmable [9] and overlay [2] networks have also benefited from advances in system and network virtualization. Network virtualization architectures are discussed in [13, 44, 8, 22, 40].

The state-of-the-art information models available to the systems and computer network domains are the Common Information Model (CIM) [17] (proposed by the Distributed Management Task Force), the Shared Information/Data (SID) model [49] (proposed by the Tele Management Forum) and the Directory Enabled Network next generation (DEN-ng) [45] (proposed by the Autonomic Communications Forum). Each of these models focuses on network aspects from a different perspective. CIM applies a holistic approach, conceptualizing computing systems and networks in general, whereas SID and DEN-ng spawn from the telecommunication industry and better represent business (as well as technical) details, in a telecommunications context. CIM has been used as the basis for the creation of

other information models that focus on specific application areas, such as virtual machines and virtual network environment provisioning [25]. An overview of the three main information models can be found in [34, 1] and their use in a network virtualization context is discussed in [28, 30]. Testbed network implementations, pursued in academic and/or private sector projects (mainly US and EU funded), introduce some level of formalization across different layers of their architecture [47]. Information models can be found in the Novi and Geysers projects with the Novi [28] and LICL [27] proposals, respectively. Finally, special mention should be given to Software Defined Networking (SDN) [42], a new, promising, paradigm in network architecture and management [31]. It has its own data model and facilitates change in the network control logic [16]. SDN enjoys broad industry support given the flexibility it offers in data center fabric management.

Two other areas of related research can be identified in the context of modeling infrastructure elements. The first area refers to the Network Description Languages (NDLs) [21], some of which contain small information models and other modeling approaches. These languages have been designed with the goal of imprinting network characteristics in a structured and hierarchical manner. Code developed in any of these languages can be used in a diverse array of applications (e.g. as input to special purpose software). Therefore, NDLs are used as modeling tools for the design and application of abstractions on physical and logical representation layers of the networking infrastructure. The second area refers to work based on the Management Information Base (MIB) concept [23], i.e. databases storing management information about devices and applications. These databases are populated and used by management applications, using specialized protocols such as the Simple Network Management Protocol (SNMP). MIBs result in data models of managed entities and this concept can be applied to abstract physical and logical device state and configuration as well as application specific information.

Hypervisor concepts in a modeling and management context are largely overlooked. Sporadic support can be found in some proposals, but it is limited in scope compared to the complexity and details involved in managing hypervisor architecture. Current information models treat hypervisors as transparent elements of the virtualization layer and begin abstraction from the virtual system or virtual network point. Partial and indirect support can only be found in CIM, in DEN-ng, as well as in MIBs. In CIM, a hypervisor (not a virtual machine) can be instantiated as a subclass via the OperatingSystem class and the built-in hypervisor virtual switch, respectively, via the UnitaryComputerSystem class. Although CIM (in the System Virtualization Model [19]) elaborates on modeling and management actions on a virtual machine and on its host computer system, it does not explicitly account for the hypervisor layer. In a similar fashion, DEN-ng could be extended via subclassing from either the PhysicalResource and LogicalResource or the VirtualSystem and VirtualImage classes. In MIBs the only relevant references are the Virtual Machine Monitoring [37] and the Virtual Machine Manager [5], both at draft status. These objects can store basic hypervisor information (list of guest virtual machines, virtual processor information and mappings of logical storage and network interfaces). Cur-

rent hypervisor technologies are very complex and incorporate several details and operational specifics that cannot be abstracted and managed by current proposals.

Choosing a modeling approach for application in a new project involves determining, beforehand, the target use of the model to be created in conjunction with the modeling capabilities offered. Three main factors affect the selection:

1. the kind of resources that need to be modeled,
2. the requirement for actual implementation in real applications (i.e., the need for instrumentation) and,
3. the capabilities of the desired model for describing the infrastructure in question.

Virtual network environments can be multidimensional entities with respect to the nature, role and function of their elements. Every aforementioned related work presents different advantages and disadvantages. From a pure modeling perspective it is clear that the three main information models are superior in modeling concepts and features than those found in other proposals. Nevertheless, these models are complex and require greater learning effort and increased development and maintenance costs. Despite of the available modelling approaches, it remains to be determined which can be best applied for abstracting the diverse nature of infrastructure resources, relations and other information, efficiently. Ultimately, choosing a model will depend on a number of factors, both modeling related and other (technical factor, cost, etc.)

3 THE PROPOSED INFORMATION MODEL

A model is an abstract representation of a real world system. Different models exist for representing the structured and unstructured information, relationships and elements in a given virtual network architecture. This diverse variety of information does not always fit into a particular type of model. In many real life cases it may be necessary to combine different approaches to reach some suitable representation. The modeling proposals discussed in the previous section cannot abstract efficiently a virtual network infrastructure based on hypervisor-provisioned resources. This is due to the missing explicit support for hypervisor entities and related characteristics. In order to account for semantic representation of such resources we propose the KF model, a CIM-based conceptual representation of the different components that constitute a virtual machine-based network. The model can cover physical and logical components supporting the virtual network along with its settings, modes of operation and statistical elements of the involved hypervisors and virtual machines. The model is extensible so as to include new elements that need to be introduced in a hardware-agnostic way. As a result, it can be applied to a wide variety of scenarios and does not depend on any particular hardware implementation. On the design aspect, the model, at the logical level, semantically incorporates a virtual network spanning with a number of virtual server hosts (acting as active network elements and provide its core resources) along with the virtualization techniques

(physical nodes, hypervisors, virtual machines – VMs) [23] employed in such design. System provisioned resources (such as CPUs, memory and I/O) as well as other relevant operational parameters are included in the model. Given the agnostic nature of the model various virtualization platforms are supported as long as proper providers are developed adhering to the CIM approach. In this context, a management server is used containing the model implementation, the data repository for the managed environment, and the resource controller for performing the necessary actions on the infrastructure. The controller is also implemented on the hosted virtual machines depending on the capabilities offered by the different virtualization platforms. The resource controller constantly communicates with the managed systems (hypervisor and hosted virtual machines), receives utilization data and issues the necessary commands as per the statistical processing. Infrastructure operators can monitor the environment and initiate further management actions. A high level overview of a managed environment using the proposed model is presented in Figure 2.

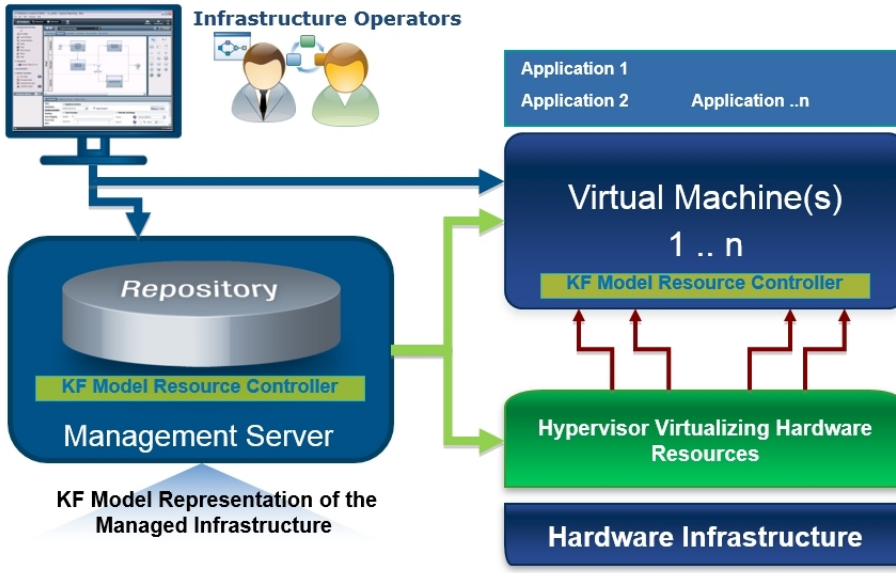

Figure 2. Management environment architecture using the proposed model

A host-based virtual network implementation, whether entry level or full scale, includes several constituents, ranging from infrastructure (hardware, software) to services and user roles. Each constituent part accesses some resource and, in certain scenarios, can be the resource owner. Furthermore, the same resources may be used via different methods and interfaces (e.g., accessing a resource via the virtualization's management interface is different than via the external client interface). The pro-

posed model addresses these issues and provides clear abstractions and methods in order to account for multiple resource ownership and varying access interfaces of the same resource. The model includes policy support pertaining to authorization (i.e., who is allowed to access the resource), conditions (based on what rules/situations the resource is accessible) and access methods (how the resource can be accessed). Policy support spans the hypervisor layer and the virtual devices it provides (e.g. the virtual switch). Semantics are also provided to support state information for all resources involved, including the hypervisor itself and the physical system it operates on. This arrangement allows for decision making based on the managed environment in which it is applied. This feature enables management applications to react based on a dynamically changing infrastructure environment (e.g. as in the cloud computing area).

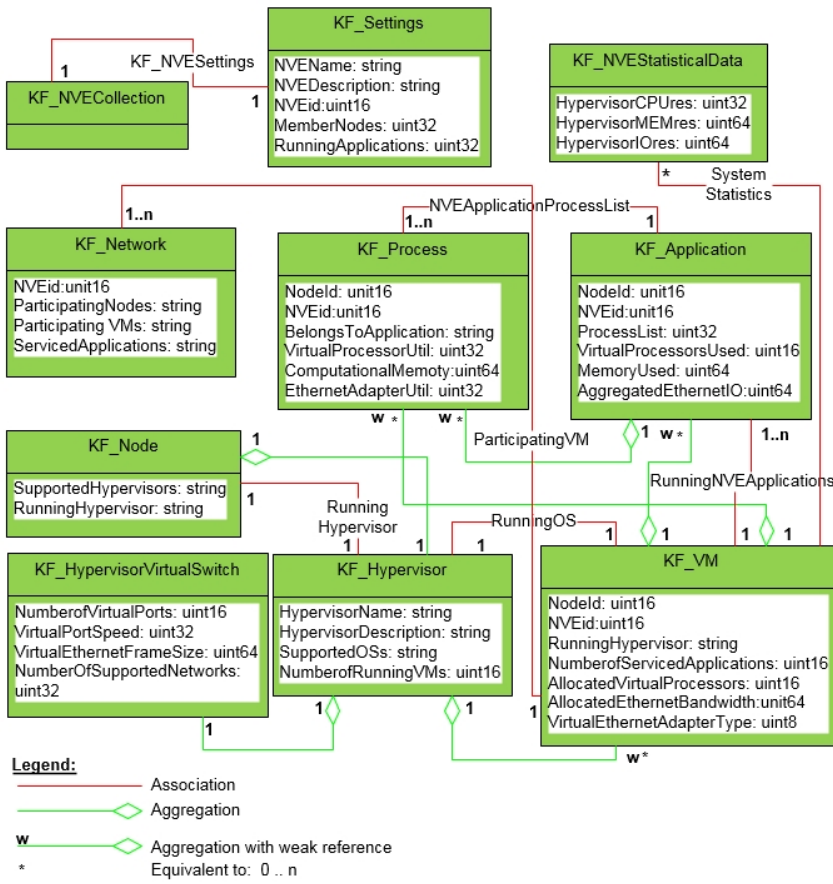

Figure 3. UML diagram of model classes, their properties and associations

The proposed model (Figure 3) consists of ten classes representing the hypervisor, virtual machine, network, configuration and virtual network architecture operation parts. These classes (Table 1) together with the extensibility characteristics of the model are adequate for basic design representation of any network employing virtual machines hosted on a hypervisor (regardless of the hypervisor selected). Additional features and facilities can be abstracted by extending the model. Hence, the need for initial class over-commitment, which would incur difficulties in the model design, is eliminated. An exclusive namespace has been applied to class naming with each class name prefixed with a *KF*. In the current version of the model the following elements of a virtual host based network are referenced: computing (systems) nodes, hypervisors, hypervisor virtual switches, virtual machines, processes, applications, virtual networks along with their settings, and statistics. All these elements are semantically represented at the logical level and the virtual network environment is conceived as a collection of the referenced entities. This approach allows for the simplification of handling and the consolidation of global characteristics, such as settings, statistics, naming, etc. The virtual machine part uses six classes that handle the physical server infrastructure as a hardware node with a running hypervisor, a number of participating VMs, and the processes and applications running on these VMs.

In the proposed model, intranode networking is partly represented by a specific *KF_HypervisorVirtualSwitch* class which details the hypervisor in-memory IEEE 802.1Q VLAN compatible virtual switch. This is modeled as a specialized CIM UnitaryComputerSystem. Networking information is also shared with the *KF_Network* class. The latter includes properties necessary for mapping a virtual host based network notion as a whole entity. A special class is used for handling statistical data, resulting in a total of three classes abstracting networking characteristics. A number of associations have been designed which, being double-ended references, return specific operational data depending on the invocation method (i.e., reporting how may virtual Ethernet adapters be allocated per VM, which is the physical node's running hypervisor, which applications operate per VM and per virtual network, etc.)

The model design allows for the inclusion of any manageable entity by implementing proper extensions. These can augment the model scope and, thus, the managing application functionality. As an example, suppose that the need arises for the handling of transaction-based performance characteristics or for the management of a virtual router instantiated by a virtual server. The first need can be tackled by a *CIM_UnitOfWork* derivative subclass [18] whereas the second need via extending the *KF_VM* class to include the required management methods. This extensibility derives from the design logic of the model and allows for the easy inclusion of new features and elements. CIM schemas are expressed in the Unified Modelling Language (UML) and their syntax description is composed in the Managed Object Format (MOF) [41]. The proposed model, leveraging CIM inheritance, provides for the following, general requirements:

Class	Purpose	CIM Parent Class
KF_Node	Describes a physical node (computing server) participating in a VNE	UnitaryComputerSystem
KF_Hypervisor	Describes the physical node Type 1 or Type 2 hypervisor	OperatingSystem
KF_Hypervisor VirtualSwitch	Describes the hypervisor in-memory IEEE802.1Q VLAN compatible virtual switch	UnitaryComputerSystem
KF_VM	Describes virtual servers running on top of the hypervisor, along with a UNIX-like operating system	OperatingSystem
KF_Process	A process running in the virtual server	UnixProcess
KF_Application	An application composed of 1..N processes	SoftwareFeature
KF_Network	Describes a virtual network	Network
KF_NVE Collection	Refers to all elements comprising an VNE	CollectionOfMSEs
KF_NVE Statistical Data	Holds statistical elements about an VNE	SystemStatisticalInformation
KF_Settings	Consolidates VNE-wide settings	SettingData

Table 1. Brief description of the classes contained in the proposed model

- Enables clients that are unaware of virtualization to manage the allocated resources of virtual systems. Resource management operations (such as add or remove CPU core) are available similarly on virtual or physical systems.
- The model is flexible and general enough to support all types of platform virtualization including hypervisor-based virtualization, logical and physical partitioning, and operating system containers. It is a matter of suitable provider instrumentation for any virtualization platform to be supported.
- Management operations are modeled in such a way that reasonable defaults are made available wherever possible. The accepted performance of the managed system is baselined and targeted decisions can be made based on that.
- The model is extensible so that new methods can be included to support further resource items or operation algorithms.

3.1 Relevance to Other Projects

The CIM virtualization model addresses the concepts of resource allocation, system virtualization and virtual devices by extending the existing CIM system modeling

and reusing system and logical device classes to model virtual systems. Our proposal is an enhancement and fills the current gap by directly addressing the hypervisor, its functional and operational specifics and the internal virtualization layer pertaining to networking (virtual switch, virtual network interfaces). As described in the previous sections, these aspects are addressed poorly, if at all, by other information models. In both the industry sector and the Open Source community, three software solutions have been made available for the purpose of providing specific management implementation options to respective management products. These efforts have not resulted in generic hypervisor models, but rather on limited CIM Schema extensions focusing on virtual machine monitoring and management for each hypervisor vendor. More specifically, Microsoft Corporation Hyper-V Windows management instrumentation provider [38] exposes a CIM-based interface, permitting users to monitor and control virtual machines hosted on a Hyper-V server. VMWare CIM SMASH/Server management application programming interface [51] is the equivalent product for the ESXi hypervisor. It allows CIM-compliant management applications to manage the hypervisor and its virtual machines. Finally, Libvirt-CIM [35] is a CIM provider for managing Linux virtualization platforms using the libvirt library. This product implements the virtualization class model from the CIM experimental schema and focuses on managing the XEN [48] hypervisor and the virtual machines hosted by it.

Extensions have already been introduced to CIM for the purpose of virtual networking management (the VNM model [20]). Among others, these cover Ethernet port resource virtualization and virtual networking components management including virtual Ethernet ports, virtual Ethernet switches, and Edge Virtual Bridging (EVB) as defined by IEEE 802.1Qbg. The focus of the VNM model is to unify the external (LAN/SAN) and the internal (virtualization platform) network management efforts under a single perspective. VNM and our proposed model differ in scope – with the latter being hypervisor and not network-centric. In the proposed model we have made the design decision to provide a distinct class for the virtual switch. Consequently, the virtualization platform specific elements are better isolated and more distinctly grouped with the hypervisor operational data. In this manner we provide a unified approach to managing the hypervisor resource, while focusing on the virtualization platform and not on the external networking infrastructure. Subclassing from VNM would be possible. However, this would incur inheriting and reusing several elements that pertain to the external network, together with the interface needed for connecting to it (i.e., the attached bridge, the network port profile database, etc.) Although the design followed introduces a new virtual switch class, it provides better encapsulation and superior control in handling the hypervisor details.

From a modeling perspective, our model (based on CIM) follows the standard CIM logic and representation methods. Representation methods vary in the literature; UML and the Extensible Markup Language (XML) are mostly used, both mature and very widely accepted standards. MIBs employ the Structure of Management Information standard (SMI). SMI includes module, object and notification definitions for describing information semantics, managed object description and

management information transmission respectively. SMI is inherently limited in semantic representation and cannot support complex data structures; hence, MIBs are bound by this restriction. This is further aggravated as MIBs use vendor specific data, added as sub-trees, thus resulting in less standardization, even though SMI has been defined as a standard. On the other hand, SID and DEN-ng use UML which is far richer in semantic capabilities and techniques than SMI and can be used to construct and support complex models. Although expressed and maintained in UML, CIM does not result in fully compliant UML models. Information and concept expression in UML leverages the advantages of XML representation. The NOVI, LICL and SDN models are exceptions to the general rule and use Owl, VXDL and YANG, respectively.

Furthermore, the main models have been designed to cover a broad scope; hence incorporate greater detail and more modeling mechanisms than the rest of the proposals. The proposed KF model, inherited from CIM, employs the same modeling mechanism. SID partially originates from an older release of DEN-ng and this has resulted in the two models sharing common concepts in model creation (relationships, attributes etc.), up to a certain extent. Both models differ from CIM in that different modeling approaches (meta-models, etc.) are used. SID and DEN-ng are pure information models whereas CIM is not, as it is not entirely technology agnostic. CIM provides for information abstraction that can be extended to include new items of the infrastructure and can be adapted to changes in management protocols. The model, however, does not include semantics for business processes and logic as do the other two main models. On the other hand, MIBs are technology-dependent data models that represent virtual containers for managed objects and their information. MIBs are much focused and limited in scope, most often abstracting the specifics of a particular device or protocol.

The three main models and the KF model are object-oriented in design, whereas MIBs are hierarchical tree views of the managed objects. This implies that extending the KF model presupposes a clear understanding of parts of this model and of CIM, their class inheritance and associations. Extensions in MIBs are done by means of adding sub-trees to the hierarchical structure; a simpler process per se, partly accounting for MIBs and SNMP popularity. Additionally, DEN-ng provides a well-designed metadata model, whereas the other two main models lack this feature. The minimal approach in class structure enables clearer design and better understanding of the model (in contrast to CIM and SID where thousands of classes are used). The NOVI information model is rich in features as it supports policy, context, capability and state. LICL provides context-aware decisions and state capturing of infrastructure resources. A more detailed discussion on the aforementioned points can be found in [34].

3.2 Experimental Implementation

In order to validate the applicability of our proposal, we introduced an actual Cloud datacenter server and network setup, with virtual networks and virtual servers pro-

cessing a particular workload. We applied the proposed model, examining how it can be used in order to describe the managed environment and whether it can be extended to facilitate some management action on that environment. The test architecture consists of an array of UNIX systems and related software (IBM pSeries 740 UNIX server divided into three virtual machines (VM) [32]), with each VM running the AIX operating system at version 7.1. Employing the IBM Power Hypervisor [4] capabilities each VM has been assigned two p7 CPUs, one GB of RAM memory and one virtual Ethernet adapters realized via the in-memory IEEE 802.1Q virtual switch. Each VM has been configured on a different virtual LAN and assigned an IP address of its own address space. The first VM acts a stress test source system, running custom code in order to hammer a web server running on the third VM. Network-wise the stress station and the web server VMs are using the second VM, which assumes the role of a network gateway. This second VM runs a copy of the "gated" routing daemon and serves as a virtual router for the 1.1.1.0/24 and 2.1.1.0/24 IP networks (for the first and third VM, respectively). Figure 4 shows a high-level representation of the experimental testbed.

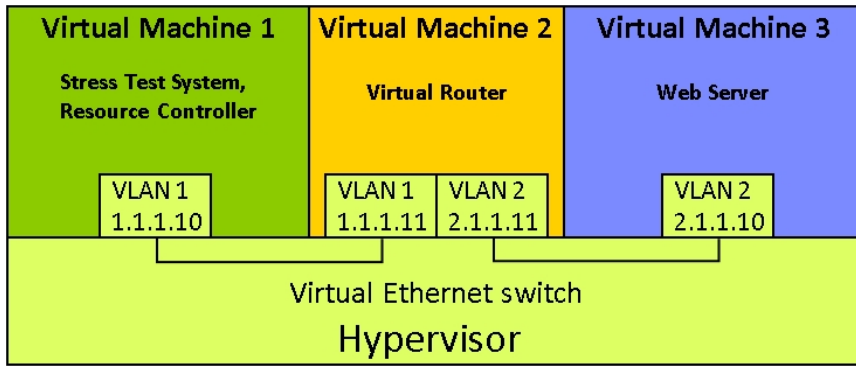

Figure 4. High-level logical depiction of the experimental cloud server and network testbed

The first VM serves an additional role as an aggregation system. It collects the performance information from the virtual router and the web server, gathering utilization patterns and producing consolidated statistics of the core resources provisioned on each VM. In addition, this VM collects hypervisor statistics for reconciliation of performance data produced at the operating system level, against data provided by the physical system's hypervisor. The KF model's features adequately describe the aforementioned test architecture and host-based virtual networks via the *KF Node, Hypervisor, HypervisorVirtualSwitch, VM, Network* classes. Virtualized hardware characteristics, such as provisioned CPU cores, memory and virtual Ethernet adapters' capacity, are contained in the respective array of class attributes (e.g. AllocatedVirtualProcessors, AllocatedEthernet-Bandwidth). In a similar fashion, resource allocation is gathered via AIX monitoring tools (sar and vmstat commands) and dynamic reconfiguration and operation commands to the p740 system's

Hardware Management Console (HMC). The resulting information is handled via the attributes VirtualProcessorsUsed, MemoryUsed and AggregatedEthernetIO. Inheritance from the CIM Network class allows for network attributes description such as VLAN ids, IP addressing and subnet masks.

Having introduced the experimental architecture and its KF model abstraction, we investigated how the model can be extended to instantiate a management approach on provisioned resources. In order to provide a realistic case, as close to actual data center infrastructure resources management as possible, we chose a statistical method known as Statistical Process Control (SPC). This method does not appear currently in the literature in the context of computing server dynamic resource allocation. SPC is an analytical method which employs statistical techniques to the monitoring and control of a process, in order to ensure that it operates within designed limits for acceptable output. Under SPC, a process behaves predictably to produce as much conforming product as possible, with the least possible waste. While SPC has been applied most frequently to controlling product manufacturing lines, it applies equally well to any process with a measurable output. SPC indicates when an action should be taken in a process, but it also indicates when no action should be taken – the latter in the case when the process is behaving as expected. Key tools in SPC are control charts which monitor processes to show how they are performing and how their capabilities are affected by changes to the process. This information is then used to make quality improvements. Control charts are also used to determine the capability (capacity) of the process. They can help identify special or assignable causes for factors that impede peak performance. Control charts offer a mechanism for continuous process monitoring and improvement. The reader is referred to [53, 55, 56] for detailed information on SPC. The KF model *KV_VM* class was extended to include the required methods, as this has been provided for in its design.

We implemented the required method functionality into the model provider in order to incorporate the logic for applying SPC on VM-based datacenter infrastructure. The provider processes the resource utilization data and determines whether corrective actions are needed. Should these be necessary, they are applied on the VM in order to bring the environment back in-control, under accepted performance parameters. Operation is performed in an online mode with the provider receiving and processing utilization data at a continuous rate during the desired management period. Thus, the provider assumes the role of a resource controller (Figure 5). The data flow described contains various operating system and model provider created messages. Listing 1 shows typical code that creates a control message for modifying the allocated CPU cores on the target VM of the managed system – VM2 in the experimental implementation. This message is issued as a response to the statistical processing of a utilization message (for space delimited values of primary CPU statistics) from the target VM.

When the CIM server handling the proposed model definitions receives queries about the instantiated model classes, the responses containing the infrastructure-related management data are provided in XML format and embed the values for

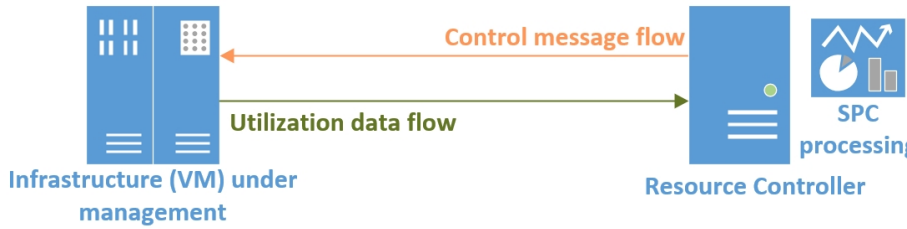

Figure 5. Logical control flow of utilization and resource reconfiguration data

```
Resource utilization control message>>
13:37:08 -- %user %sys %wait %idle physc %entc lbusy app vcsw phint
47.1 20.6 2.3 29.9 8.15 108.7 24.2 1.32 11562 826

Control execution>>

# Managed system name
FRAME=Server1-SN06C0121
# Hardware Management Console name
HMC=hmc1
# How much CPU would be added
PROC_TO_ADD=1.6
# Minimum processing units required for each virtual processor: 0.10
VPROC_TO_ADD=16
# VM where resource will be added
NODE_TO_ADD=SRVCBP1

echo "Adding $PROC_TO_ADD processor(s) to $NODE_TO_ADD"
# Message creation and VM control
ssh hscroot@${HMC} "chhwres -r proc -o a -p $NODE_TO_ADD -m $FRAME
--procunits $PROC_TO_ADD --procs $VPROC_TO_ADD -w 1"
```

Listing 1. Example of resource modification control sequence for adding CPUs

each particular variable of the class in question (Listing 2). These messages can be parsed by the management application so that infrastructure-related decisions are made.

Following the extension of the model and the implementation of the required code, the virtual router and the web server VMs were load-stressed using the Apache Foundation *jmeter* [29] and the IBM *nstress* [39] software packages. Dynamic VM CPU reconfiguration was attempted on the virtual router VM via the implemented SPC methods. Prior to applying the methods, several test runs were executed for which detailed performance data was collected. This data was analyzed and the required statistical attributes were calculated, because it is necessary to discover

```
<?xml version="1.0" encoding="utf-8"?>
<CIM CIMVERSION="2.0" DTDVERSION="2.0">
<MESSAGE ID="1566" PROTOCOLVERSION="1.0">
<SIMPLERSP>
<INSTANCENAME CLASSNAME="KF_VM">
<KEYBINDING NAME="AllocatedVirtualProcessors">
<KEYVALUE VALUETYPE="uint16">
5
</KEYVALUE>
</KEYBINDING>
<KEYBINDING NAME="NumberOfServicedApplications">
<KEYVALUE VALUETYPE="uint16">
8
</INSTANCENAME>
</SIMPLERSP>
</MESSAGE>
</CIM>
```

Listing 2. Example of XML response message sample

the proper control limits, UCL, CL and LCL [54] before the SPC method can be applied. A new three-hour test run was then executed with the virtual router VM CPU capacity initially decreased to one core so as to enforce and simulate a lack-of-resources condition. The imposed workload was increased by 100 % every 30 minutes, for 5 consecutive cycles, before being restored to the initial stress level. The KF controller successfully managed the VM, dynamically modifying its CPU capacity as per the observed CPU utilization changes, adapting to the varying demand for resources. The VM remained in controlled operation, avoiding poor service provisioning due to resources starvation. The extra CPU cores were released once the workload had decreased. The VM remained in controlled operation, avoiding poor service provisioning due to resources starvation. Graphical results of the final test run are presented in Figure 6.

4 DISCUSSION

Concerning system virtualization, the test architecture's operation can be characterized by the utilization of the core server resources (CPU, memory, I/O) and several actions can be based on this (SLA monitoring, accounting, VM operations based on resource consumption, etc.) We implemented a typical infrastructure configuration, as found in most cloud datacenters, and used the KF model to semantically abstract the involved management information. Then we proceeded to demonstrate the model capabilities by extending it to include a statistical approach for dynamic resource allocation. The KF model was successful in supporting both undertakings. The outcome was to measure the utilization of infrastructure resources so that

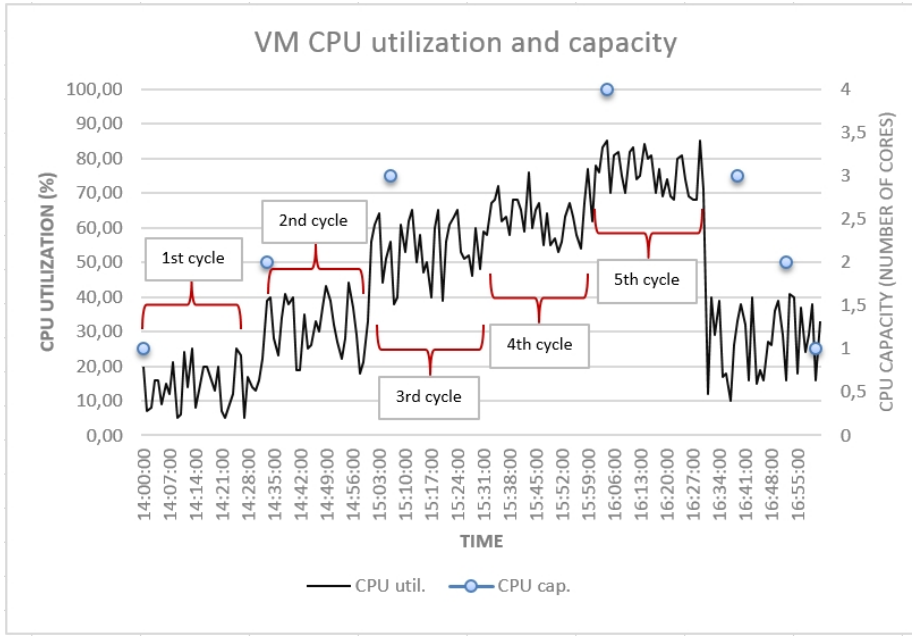

Figure 6. Virtual router VM CPU utilization and capacity over time

proper decisions can be based on those measurements, and instrumented via the model (e.g., to migrate a node or add resources to it if utilization reaches a certain level, or trigger SLA alert if the resources' consumption exceeds the agreed limit). The KF model abstracts the aforementioned utilization metrics with the use of providers designed for this task, running on each VM. The model is not limited to these metrics or scenario and can be extended to include other situations, as long as these can be instrumented by the providers at the VM level. If a virtual network spans more than one VM or extends beyond the VMs operating on a single hypervisor, then it is necessary to introduce an aggregation system (AS) where the management application will reside. The CIM client running on this system will collect the management data from the VMs and perform the necessary calculations as per the applied scenario. In the case of multiple VMs running on a single hypervisor it is possible to host a CIM client on one of the VMs and monitor the resources at the hypervisor level. This could be desirable in some scenarios, as it may produce more meaningful results. After all, hypervisor-level monitoring is more accurate as it avoids the overhead imposed by operating system and/or monitoring software.

SPC can be applied to the management of specific elements of the test architecture, in particular, to VM hardware capacity. VM resources (e.g. CPU, memory and I/O) can be monitored and dynamically tuned, if the operating system and

the hardware permits it. For example, in SLA-constrained virtual network, VM reaching a sustained CPU utilization of 80 % can be allocated extra CPUs, in order to continue operating as expected. Within the CIM context, SPC cannot be directly mapped as it is not a manageable entity. It can, however, be indirectly instrumented via proper methods in specific classes that monitor and manage CIM-handled resources using SPC techniques. The new methods, presented in Table 2, are implemented in the KF provider and are specific for the operating system and hardware. Much of the power of this CIM-based SPC control lies in the ability to examine the variation of the process in real-time allowing for adaptive actions. This approach can be applied, via suitable model extensions and the introduction of new methods, to any managed resource. Variations in the process that may affect the quality of the end product or service can be detected and handled, reducing the probability of error in a well-designed and tuned environment.

Method	Purpose	Usage Scope
SetControlLimits	Calculates the Lower and Upper Control Limits based on the observed CPU utilization.	Any VM performance fluctuations outside these limits may result from common causes inherent to the system, such as normal VM imposed load. These normal fluctuations are attributed to statistical variability.
CheckStatus	Checks whether the VM's performance is within acceptable operating parameters for the anticipated and designed load. Implements a control chart equivalent and allows for live pattern analysis.	The variance of the VM performance is compared over time, against the upper and lower control limits to see if it fits within the expected variation levels.
ModCPUs	Interfaces with the hosting hypervisor in order to add or remove CPUs.	Modifies the allocated CPU resources so as to bring the VM in a stable and accepted performance state. This is facilitated by calls to the hypervisor control interface.

Table 2. Statistical Process Control methods built in the KF_VM class

Our proposed model addresses in detail the specifics of the virtualization layer and the concept of hypervisors. The latter has been overlooked in all available models and no proper abstraction has been proposed. Some of the current models reflect on this matter only superficially. Only recently has this need been acknowledged, indirectly in [15]. More specifically, the authors address the problem introduced in management operations given the use of different hypervisor types in the infrastructure. They accurately state that *management attributes are the foundation of any*

management operation and *without current information about the managed objects, sensible and useful management actions cannot be executed*. The authors provide an analysis of virtual machine attributes and their representation across different hypervisors, in parallel with a possible CIM-based abstraction for each. Although this analysis focused on the virtual machine side and did not attempt to model the hypervisor, it provides a clear indication of the complexity and the detail involved in environments where hypervisors are used if you wish to be engaged in modeling. Our proposal ensures that any specific hypervisor platform can be supported (vendor independent and implementation agnostic), providing the ability to abstract and manage virtualized network connectivity within both a single and cross-hypervisor architectures. This advantage is more pronounced in environments where shifting workloads among nodes (i.e., rerouting traffic across hypervisors) is the basic characteristic. Aspects of systems virtualization and physical/logical server resources and characteristics cannot be ignored as they influence the operation of the network and, ultimately, the quality of the provided service [36]. Hypervisors [24] do impose a new layer of virtualization by creating a secondary address space intervening between network applications and physical infrastructure layers [4, 33]. Furthermore, advances in other areas result in new developments that need to be taken into account. The complete virtualization of the network address space is still a relatively new field, influenced by the use of hypervisors in the network infrastructure stack and by the IPv6 evolution. These technologies are now offered by all server virtualization products e.g. VMware [52], IBM [32], Oracle [43] with the vSphere, PowerVM and Cross-bow/SPARC/Oracle VM hypervisors, respectively, and the XEN hypervisor [48].

The proposed information model provides a clear and flexible solution to the problem of the lack of hypervisor support in other modeling proposals. Aspects of systems virtualization and physical/logical server's resources and characteristics cannot be ignored as they influence the operation of the network and, ultimately, the quality of the provided service. Hypervisors do impose a new level of virtualization by creating a secondary address space intervening between network applications and physical infrastructure layers. Therefore, by applying the proposed model it is easier to *efficiently* manage the virtual network environments apparent in modern cloud data centers. Our proposal provides a conceptual view of the managed environment, that attempts to unify and extend existing instrumentation and management standards (SNMP, DMI, CMIP, etc.) using object-oriented constructs and design. Following the specifications of the parent CIM model, our proposal does not require any particular instrumentation or repository format: it allows to unify the data, using an object-oriented format, made available from any number of sources. This very object-oriented nature reduces the complexity of the problem domain, as high level and fundamental concepts (the "objects" of the management domain) are defined. Furthermore, dependencies, component and connection associations can easily be depicted and handled, thus enabling problem determination and root cause analysis. To illustrate this, consider a troubleshooting scenario where the infrastructure administrator discovers that one virtual network

card in a virtual machine is at high, or in a bottleneck utilization. Investigating further (by traversing the proposed model's associations and exploring additional subclasses and data objects), the administrator finds that the hypervisor-provisioned port of that card has a very high traffic rate, and its traffic is related to a particular application. Again, following associations, the administrator can determine the "owner" of the system currently attached to the network port. If the owner cannot be reached, the administrator can use a standard method (instrumented in the KF provider) to "add another network port" to the VM and balance the traffic, thereby returning the hub to normal operating levels and restoring proper service operation.

5 CONCLUSIONS

New technologies and trends have changed or influenced the way people connect to and use computer networks. Innovative methods for network connection and information access have been provided, nevertheless leading to increased complexity of relevant implementations. Technologies such as wireless networking, mobile telephony, optical networks and cloud computing allow users to connect to a network from nearly any geographical location, using a variety of devices in order to access the provided services. Recent technological advancements complicate network architecture and operation, introducing new aspects that affect network management. Cisco Systems projects that thirty seven billion intelligent devices (some of peculiar nature: smart fabrics, pills, etc.) will connect to the Internet by 2020 – dramatically increasing the operational complexity of the *The Internet of Everything* [57]. The complexity of virtual network implementations tends to increase, resulting in impediments in the assurance of quality of service, monitoring, and administration, on an end-to-end basis, unless some standardization approach is applied.

In this paper we introduced KF, a CIM-based information model, facilitating the description and management of virtual networks where computing servers and hypervisors are used as active network elements. Our approach was successfully demonstrated on a typical Cloud data center virtual network setup, allowing for the conceptual representation of involved components and the introduction of targeted actions. We have provided a tool towards the more efficient management of these, heavily virtualized, infrastructures. This is essential in the modern networking landscape which requires an increased level of standardization, end-to-end, from the abstraction down to the management side. Our ongoing work focuses on enhancing the proposed model with semantics for different server and network hardware characteristics, as well as increasing the level of detail in those already included in the design. In addition, we will investigate further the application of statistical process control and its capability for enabling workload pattern-adaptive infrastructure automation.

REFERENCES

[1] Agoulmine, N.: Autonomic Network Management Principles. Elsevier Academic Press, Amsterdam, The Netherlands, 2010.

[2] Andersen, D.—Balakrishnan, H.—Kaashoek, F.—Morris, R.: Resilient Overlay Networks. ACM SIGOPS Computer Communication Review, Vol. 32, 2001, No. 1, pp. 66–66, doi: 10.1145/502034.502048.

[3] Armbrust, M.—Fox, A.—Griffith, R.: Above the Clouds: A Berkeley View of Cloud Computing. Technical Report UCB/EECS 28, University of California, Berkeley. Available on: `http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html`, 2014.

[4] Armstrong, W.—Arndt, R.—Boutcher, D.: Advanced Virtualization Capabilities of POWER5 Systems. IBM Journal of Research and Development, Vol. 49, 2005, No. 4, pp. 523–532, doi: 10.1147/rd.494.0523.

[5] Asai, H.—Sekiya, Y.—Shima, K.—Esaki, H.: Management Information Base for the Virtual Machine Manager. IETF Draft, 2013. Available on: `http://tools.ietf.org/html/draft-asai-vmm-mib-00`, 2014.

[6] Baroncelli, F.—Martini, B.—Castoldi, P.: Network Virtualization for Cloud Computing. Annals of Telecommunications, Vol. 65, 2010, No. 11-12, pp. 713–721, doi: 10.1007/s12243-010-0194-y.

[7] Botta, A.—Canonico, R.—Stasi, G.—Pescape, A.—Ventre, G.—Fdida, S.: Integration of 3G Connectivity in PlanetLab Europe. Mobile Networks and Applications, Vol. 15, 2010, No. 3, pp. 344–355, doi: 10.1007/s11036-010-0224-z.

[8] Boucadair, M.—Georgatsos, P.—Wang, N.: The AGAVE Approach for Network Virtualization: Differentiated Services Delivery. Annals of Telecommunications, Vol. 64, 2009, No. 5, pp. 277-288, doi: 10.1007/s12243-009-0103-4.

[9] Campbell, A. T.—Meer, H. G. D.—Kouvaris, M. E.—Miki, K.—Vicente, J. B.—Villela, D.: A Survey of Programmable Networks. ACM SIGCOMM Computer Communication Review, Vol. 29, 1999, No. 2, pp. 7–23, doi: 10.1145/505733.505735.

[10] Canonico, R.—Emma, D.—Ventre, G.: An XML Description Language for Web-Based Network Simulation. Proceedings of the 7th IEEE International Symposium on Distributed Simulation and Real-Time Applications, Delft, The Netherlands, 2008, pp. 76–81.

[11] Chowdhury, N. M. M. K.—Boutaba, R.: A Survey of Network Virtualization. Computer Networks, Vol. 54, 2010, No. 5, pp. 862–876, doi: 10.1016/j.comnet.2009.10.017.

[12] Chowdhury, N. M. M. K.—Boutaba, R.: Network Virtualization: State of the Art and Research Challenges. IEEE Communications Magazine, Vol. 47, 2009, No. 7, pp. 20–26, doi: 10.1109/mcom.2009.5183468.

[13] Chowdhury, N. M. M. K.—Zaheer, F.—Boutaba, R.: iMark: An Identity Management Framework for Network Virtualization Environment. Proceedings of the IFIP/IEEE International Symposium on Integrated Network Management, New York, USA, 2009, pp. 335–342, doi: 10.1109/inm.2009.5188833.

[14] Creeger, M.: Moving to the Edge: A CTO Roundtable on Network Virtualization. Communications of the ACM, Vol. 53, 2010, No. 8, pp. 55–62, doi: 10.1145/1787234.1787251.

[15] Danciou, V. A.: Bottom-Up Harmonisation of Management Attributes Describing Hypervisors and Virtual Machines. Proceedings of the 5th International DMTF Alliance Workshop, Paris, France, 2011, pp. 1–10, doi: 10.1109/svm.2011.6096460.

[16] Devlic, A.—John, W.—Skoldstrom, P.: A Use-Case Based Analysis of Network Management Functions in the ONF SDN Model. Proceedings of the European Workshop on Software Defined Networking (EWSDN), Darmstadt, Germany, 2012, pp. 85–90, doi: 10.1109/ewsdn.2012.11.

[17] Distributed Management Task Force: Common Information Model. Technical Report DSP0004. Available on: http://www.dmtf.org/standards/cim, 2014.

[18] Distributed Management Task Force: CIM Metrics Model. Technical Report DSP0141. Available on: http://www.dmtf.org, 2014.

[19] Distributed Management Task Force: CIM System Virtualization Model. Technical Report DSP2013. Available on: http://www.dmtf.org, 2014.

[20] Distributed Management Task Force: Virtual Networking Management. White paper DSP2025 version 1.0.0. Available on: http://www.dmtf.org, 2014.

[21] ERCIM: Network Description Tools and Standards. ERCIM News, Vol. 77, 2009, No. 1, pp. 33–34.

[22] Feamster, N.—Gao, L.—Rexford, J.: How to Lease the Internet in Your Spare Time. ACM SIGCOMM Computer Communication Review, Vol. 37, 2007, No. 1, pp. 61–64, doi: 10.1145/1198255.1198265.

[23] Fenner, B.: MIB Index. ICSI Networking and Security Group. Available on: http://www.icir.org/fenner/mibs/mib-index.html, 2014.

[24] Fenn, M.—Murphy, M.—Martin, J.—Goasguen, S.: An Evaluation of KVM for Use in Cloud Computing. Proceedings of the 2nd International Conference on the Virtual Computing Initiative (ICVCI), Durham, NC, USA, 2008.

[25] Fuertes, W.—Lopez, J. E. V.—Meneses, F.—Galan, F.: A Generic Model for the Management of Virtual Network Environments. Proceedings of the IEEE Network Operations and Management Symposium (NOMS), Osaka, Japan, 2010, pp. 813–816, doi: 10.1109/noms.2010.5488367.

[26] Galan, F.—Fernandez, D.—Fuertes, W.—Gomez, M.—Vergara, J.: Scenario-Based Virtual Infrastructure Management in Research and Educational Testbeds with VNUML. Annals of Telecommunications, Vol. 64, 2010, No. 5-6, pp. 305–323.

[27] Garcia-Espin, J. A.—Riera, J. F.—Lopez, E.: Functional Description of the Logical Infrastructure Composition Layer (LICL). EU GEYSERS Project Technical Report D3.1. Available on: http://www.geysers.eu, 2014.

[28] Grosso, P.—van der Ham, J.—Steger, J.: Information Models for Virtualized Architectures. EU NOVI Project Technical Report NOVI-D2.1-11.02.28-v3.1. Available on: http://www.fp7-novi.eu/deliverables/doc_download/25-d21, 2014.

[29] Apache Software Foundation: Jmeter. Available on: http://jmeter.apache.org/, 2015.

[30] van der Ham, J.—Chrysa, P.—Jozsef, S.: Challenges of an Information Model for Federating Virtualized Infrastructures. Proceedings of the 5th International DMTF Academic Alliance Workshop on Systems and Virtualization Management: Standards and the Cloud, Paris, France, 2011, pp. 1–6.

[31] Kim, H.—Feamster, N.: Improving Network Management with Software Defined Networking. IEEE Communications Magazine, Vol. 51, 2013, No. 2, pp. 114–119.

[32] IBM: Server Virtualization with IBM PowerVM. Available on: http://www-03.ibm.com/systems/power/software/virtualization/, 2014.

[33] IBM: Virtual I/O Server. Available on: http://www-304.ibm.com/support/customercare/sas/f/vios/home.html, 2014.

[34] Kontoudis, D.—Fouliras, P.: A Survey of Models for Computer Networks Management. International Journal of Computer Networks & Communications, Vol. 6, 2014, No. 3, pp. 157–176, doi: 10.5121/ijcnc.2014.6313.

[35] LibVirt Community: Libvirt CIM Provider. Available on: http://libvirt.org/CIM/intro.html, 2014.

[36] Lu, J.—Makhlis, L.—Chen, J.: Measuring and Modeling the Performance of the Xen VMM. Proceedings of the International CMG Conference, Osaka, Japan, 2006, pp. 621–628.

[37] McFaden, M.—Schoenwaelder, J.—Tsou, T.—Zhou, C.: Definition of Managed Objects for Virtual Machines Controlled by a Hypervisor. IETF Draft. Available on: http://tools.ietf.org/html/draft-schoenw-opsawg-vm-mib-01, 2014.

[38] Microsoft: Hyper-V WMI Provider. Available on: http://msdn.microsoft.com/en-us/library/cc136992%28v=vs.85%29.aspx, 2014.

[39] IBM developerWorks: Power Systems nstress Performance Tool. Available on: http://www.ibm.com/developerworks/, 2015.

[40] Oberle, K.—Kessler, M.—Stein, M.—Voith, T.—Lamp, D.—Berger, S.: Network Virtualization: The Missing Piece. Proceedings of the IEEE 13th International Conference on Intelligence in Next Generation Networks, Bordeaux, France, 2009, pp. 1–6, doi: 10.1109/icin.2009.5357110.

[41] Open Management Group: Meta Object Facility (MOF). Available on: http://www.omg.org/mof, 2014.

[42] Open Networking Foundation: Software-Defined Networking: The New Norm for Networks. ONF White Paper. Available on: https://www.opennetworking.org, 2014.

[43] Oracle Corporation: Application-Driven Virtualization, Oracle VM Server for x86. White Paper. Available on: http://www.oracle.com/us/technologies/virtualization/oraclevm, 2014.

[44] Schaffrath, G.—Werle, C.—Papadimitriou, P.: Network Virtualization Architecture: Proposal and Initial Prototype. Proceedings of the 1st ACM Workshop on Virtualized Infrastructure Systems and Architectures, Barcelona, Spain, 2009, pp. 63–72, doi: 10.1145/1592648.1592659.

[45] Strassner, J.: DEN-ng: Achieving Business-Driven Network Management. Proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS), Colorado Springs, USA, 2002, pp. 753–766, doi: 10.1109/noms.2002.1015622.

[46] Tennenhouse, D. L.—Wetherall, D. J.: Towards an Active Network Architecture. ACM SIGCOMM Computer Communication Review, Vol. 37, 2007, No. 5, pp. 81–94, doi: 10.1145/1290168.1290180.

[47] Sherwood, R.—Gibb, G.—Yap, K. K.—Appenzeller, G.—Casado, M.—McKeown, N.—Parulkar, G. M.: Can the Production Network Be the Testbed? Proceedings of the 9th USENIX Symposium on Operating Systems Desing and Implementation (OSDI), Vancouver, Canada, 2010, pp. 1–6.

[48] The Linux Foundation: XEN Hypervisor. Available on: http://www.xenproject.org/, 2014.

[49] TMForum: Information Framework (SID). Available on: http://tmforum.org/InformationFramework/1684/home.html, 2014.

[50] Tutschku, K.—Tran-Gia, P.—Andersen, F. U.: Trends in Network and Service Operation for the Emerging Future Internet. International Journal for Electronics and Communications, Vol. 62, 2008, No. 9, pp. 705–714, doi: 10.1016/j.aeue.2007.09.002.

[51] VMware: VMware CIM SMASH/Server Management API for ESXi 5.0. Available on: http://pubs.vmware.com/vsphere-50/index.jsp, 2014.

[52] VMware: VMware vSphere Hypervisor. Available on: http://www.vmware.com/products/vsphere-hypervisor, 2014.

[53] Wetherill, B.—Brown, W.: Statistical Process Control: Theory and Practice. Chapman and Hall/CRC Press, London, UK, 1991.

[54] Wheeler, D. J.—Chambers, D. S.: Understanding Statistical Process Control. SPC Press, Knoxville, Tennessee, USA, 2010.

[55] Woodall, W. H.: Controversies and Contradictions in Statistical Process Control. Journal of Quality Technology, Vol. 32, 2000, No. 4, pp. 341–350.

[56] Woodall, W. H.—Montgomery, D. C.: Research Issues and Ideas in Statistical Process Control. Journal of Quality Technology, Vol. 31, 1999, No. 4, pp. 376–387.

[57] Dave, E.: How the Internet of Everything Will Change the World for the Better. Cisco Corporation. Available on: http://blogs.cisco.com/ioe/beyond-things-the-internet-of-everything-takes-connections-to-the-power-of-four, 2014.

Dimitris Kontoudis is Ph.D. graduate of the University of Macedonia, Greece, and Deputy Director at the Information Technology Division of the National Bank of Greece. He holds his B.Sc. in physics and scientific computing (University of Greenwich, UK) and M.Sc. in advanced methods in computer science (University of London, UK). His research interests include systems/network virtualization and data centre infrastructure performance management.

Panayotis Fouliras is Assistant Professor of computer networks at the Department of Applied Informatics, University of Macedonia, Greece. He has published over 40 articles in international and national scientific journals and conferences, mainly on applied informatics and computer networks related topics. He has participated in more than 10 international and national funded R & D projects. He is a member of the InfoSec research group of the MSN lab at the University of Macedonia, Greece.