A Network-Oriented Survey and Open Issues in Cloud Computing

Luigi Atzori[1], Fabrizio Granelli[2], Antonio Pescapè[3]

[1] DIEE (Dept. of Electric and Electronic Engineering) - University of Cagliari, Italy,
E-mail: l.atzori @diee.unica.it
[2] DISI (Dept. of Information Engineering and Computer Science) - University of Trento, Italy,
E-mail: granelli @disi.unitn.it
[3] DIS (Dept. of Computer Science and Systems) – University of Napoli Federico II
E-mail: pescape@unina.it

Abstract

Cloud computing represents an emerging paradigm in the framework of ICT, describing a computing model where business applications are allocated to a combination of connections, and software and services are accessed through a web browser over a network, known as "The Cloud". This permits access to power computing through a variety of entry points and eliminates the need for organizations to install and run heavy duty applications on their computers; the data and software themselves are retrieved "on demand" like a utility service. In general, no unified and quantitative approach to analyze and design cloud computing architectures is available in the literature, as well as specific supporting technologies to control the QoS and ensure predictable performance.

The chapter is aimed at studying networking solutions for “engineering” Cloud Computing architectures, with the final goal to provide a framework for enabling analysis, design and performance improvement of such architectures.

I. Introduction

Cloud computing represents an emerging paradigm in the framework of ICT, describing a computing model where business applications are allocated to a combination of connections, and software and services are accessed through a web browser over a network, known as "The Cloud" [8]. This enables access to power computing through a variety of entry points and eliminates the need for organizations to install and run heavy duty applications on their computers; the data as well as software itself is retrieved "on demand" like an utility service [42, 43, 44]. As a matter of fact, cloud computing constitutes a specialized distributed computing paradigm, with the following features:

1. it is massively scalable and provides economy of scale;
2. it can be encapsulated as an abstract entity offering different level of service to customers outside the Cloud (i.e. the Internet, or any communication network);
3. services can be dynamically configured (e.g. via virtualization) and delivered on-demand;
4. it supports energy saving.

As in several well-known cases (such as peer-to-peer), diffusion of this paradigm was driven by software developers and service providers rather than research, leading to the existence of already available solutions - including Google App Engine [11] and Google Docs, Yahoo! Pipes [40], web Operating Systems (G.ho.st, etc.), without the support from research studies related to architecture and protocol design, performance analysis, dimensioning, and similar issues.

Indeed, the main open challenges related to the cloud computing paradigm include:

1. data portability, lack of trust and privacy issues;
2. QoS (Quality of Service) control or guarantee, as Clouds will grow in scale and number of users and resources will require proper management and allocation;

3. increase of data-intensive applications will put a heavy burden on the communication infrastructure;
4. difficulty in fine-control over resources monitoring, as the layered architecture makes it difficult for an end user (but also for a developer or an administrator) to deploy his own monitoring infrastructure;
5. virtualization itself represents both an advantage (it provides the necessary abstraction to unify fabric components into pool of resources and resource overlays) as well as a disadvantage (reliable and efficient virtualization is required to meet SLA requirements and avoid potential performance penalties at application level).

To the best of the knowledge of the authors, in general, no unified and quantitative approach to analyze and design cloud computing architectures as well as specific supporting technologies to enable to control the QoS and ensure predictable performance are available in the literature.

In this framework, this chapter is aimed at identifying and analyzing solutions for engineering Cloud Computing architectures, with the final goal to provide a description of open issues, available solutions and ongoing activities on the subject.

The chapter is organized as follows. Section II presents a brief overview to cloud computing and related issues; Section III illustrates the main network-oriented challenges related to this new computing model, while Section IV concludes the chapter with final remarks.

II. A brief view at Cloud Computing

As cloud computing encompasses several disciplines related to ICT, it is not possible to provide a comprehensive state-of-the-art. Therefore, in the following, we only provide a review of the main subjects within this wide topic.

Cloud Computing Architectures

The paper by Buyya, Yeo and Venugopal [38] presents a modern vision of computing based on Cloud computing. Different computing paradigms promising to deliver the vision of computing utilities are presented, leading to the definition of Cloud computing and the architecture for creating market-oriented Clouds by leveraging technologies (such as VMs). Chappell [5] provides an introduction on cloud computing, starting from a brief classification of the current implementations of cloud computing, identifying three main categories. Wang and Von Laszewski [20] reveal how the Cloud computing emerges as a new computing paradigm which aims to provide reliable, customized and QoS guaranteed dynamic computing environments for end-users. Their paper analyzes the cloud computing paradigm from various aspects, such as definitions, distinct features, and enabling technologies. It considers the Hardware as a Service (HaaS), the Software as a Service (SaaS) and the Data as a Service (DaaS) visions, and the main functional aspects of cloud computing. While these works provide different definitions and visions of the Cloud, all rely on a common architecture which is shown in Figure 1 and that is made of the following components: the Fabric Layer which encompasses the hardware resources; the Unified Layer, which is aimed at making available to the upper layer a uniform set of services to make use of the hardware resources; the Platform, which is the operating system for the management of the system; and the Application layer, which includes the applications provided by the Cloud.

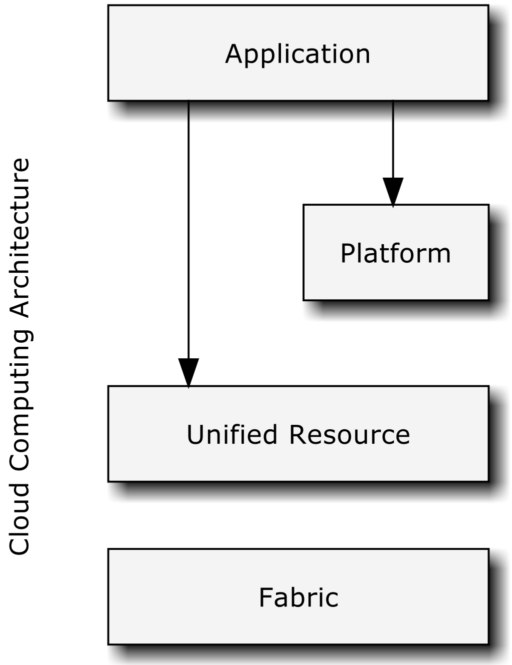

Figure 1: A reference Cloud Computing layered model.

The main distinctions between Cloud Computing and other paradigms (such as Grid computing, Global computing, Internet Computing), are the followings:

1. User-centric interfaces.
2. On-demand service provisioning.
3. QoS guaranteed offer.
4. Autonomous System.
5. Scalability and flexibility.

Cross-layer signaling

Tighter cooperation among protocols and functionalities at different layers of the protocol stack (especially between application-level clouds and networking infrastructure) is envisaged as an enabling framework for (i) supporting fine control over network and computation resources, and (ii) providing APIs for network-aware applications.

As far as cross-layering is concerned, several cross-layer approaches have been proposed in the literature so far, focusing on specific problems, providing ad-hoc solutions and rising relevant issues concerning implementation of different solutions within the TCP/IP protocol reference model. Indeed, coexistence and interoperability represent a central issue, especially in the cloud computing scenario, leading to the need for a common cross-layer signaling architecture. Such an architecture should provide the implementation of the cross-layer functionalities as well as a standard way for an easy introduction of cross-layer mechanisms inside the protocol stack. Kliazovich et al. [17] present some possible cross-layer signaling architectures that can be proposed based on ideas available in the literature, even if it must be underlined that none is currently available in the form of an enabling framework to study and fully exploit the advantages of a reasonably wide range of cross-layer solutions.

Services and Traffic analysis and classification

Service characterization [12, 23] and the related traffic classification and analysis represent important steps in the entire framework of service management and delivery [41]. Being the service characterization phase depending on the particular service scenario, low level view (network traffic level) permits to adopt well established techniques proposed in literature and at the same time a preferred way to analyze the behavior of the applications. Traffic classification represents a powerful tool for understanding the characteristics of the traffic relying on the network. Today the classification approaches used are port-based and payload-based. Such techniques were initially considered very reliable, such to be used to build reference data in the evaluation of novel classification approaches [21, 39]. Because of the increasing problems (e.g. privacy or unreliability), in the last years researchers have proposed several classification techniques that do not need access to packets content, while they are commonly based on the statistical analysis of traffic patterns and on machine-learning. The explosion of high-quality scientific literature in this field [25, 24, 35, 18] testifies the great interest in researching novel and accurate techniques for traffic classification, which find application in several networking areas. It has been demonstrated that statistical and machine-learning techniques can achieve high degrees of accuracy, and that they appear to be the most promising approaches to face problems like protocol obfuscation, encapsulation, and encryption. Despite the great effort in traffic classification and analysis, the literature lacks the studies considering traffic generated by cloud computing applications.

QoS technologies and management issues

QoS management has to be provided in a consistent and coordinated fashion across all layers of enterprise systems, ranging from enterprise policies, applications, middleware platforms, and down to network layers. In addition, a comprehensive set of QoS characteristics in categories like performance, reliability, timeliness, and security must be managed in a holistic fashion. This is the approach that has been followed by most significant works in this field since the last few years. Efforts are being made in various areas, for example, SOA, application-oriented networking, and autonomic computing [14], to achieve a scalable, secure, and self-managed service delivery framework to shorten the time-to-market of new Internet applications, as well as lower the management costs of service provider [4]. A prototype for QoS management to support Service Level Management for global enterprise services is presented in [36]. These works try to define a solution with a holistic view of the QoS management problem, starting from an architectural point of view and identifying standards, analyzing interoperability issues and defining workflows. Other studies focus more on the networking issues, that is, how service differentiation is fulfilled while transmitting the content from an end to another end of the network on the basis to high-level QoS targets. [16] specifically focuses on the issues and complexities on merging WDM and IP technologies and concentrate on making the core network efficient for transporting differentiated service traffic, adaptive to changes in traffic patterns and resilient against possible failures. [1] deals with the bandwidth management in NGN with particular attention to the Differentiated-Service-aware Traffic Engineering in Multiprotocol Label Switching networks.

Virtualization

Virtualization mechanisms [31] find the right employment in the cloud computing context. The Linux VServer [22] technology is a soft partitioning concept based on Security Contexts, which permits the creation of many independent Virtual Private Servers (VPS) that run simultaneously on a single physical server at full speed, efficiently sharing hardware resources. FreeBSD Jail [9] is the virtualization mechanism used in Emulab in order to support multiple experiments running concurrently on the same physical node. OpenVZ [30] is an operating system-level virtualization technology built using GNU/Linux. It gives the ability to run multiple isolated system instances, called Virtual Private Servers or Virtual Environments. Xen [3] is a paravirtualization system developed by the University of Cambridge. Xen provides a VM monitor for x86 processors that supports execution of multiple guest operating systems at the same time.

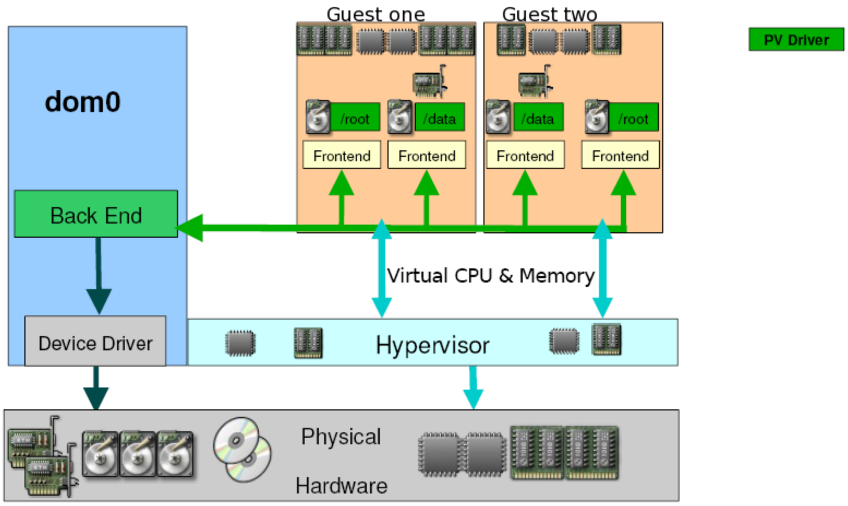

Figure 2: Xen Paravirtualization Architecture (source: RedHat).

VIOLIN [33] is a shared distributed infrastructure formed by federating computation resources from multiple domains. Usher [37] is a VM management system designed to impose few constraints upon the computing environment under its management.

III. Research Challenges for Engineering Cloud Computing Architectures

Focus of the chapter is the identification and analysis of solutions for engineering Cloud Computing architectures, with the final goal to provide a description of open issues, available solutions and ongoing activities on the subject. The term “Cloud” refers to the characteristic of accessing data and services from different access technologies through a transparent network (i.e. the Internet). The term "engineering" defines the novelty of the approach, and clearly identified the focus on the novel perspective of enabling quantitative analyses and effective design of solutions covering issues related to the support of cloud computing services, including protocols and architectures, QoS/QoE, interoperability, SLAs. As a consequence, the chapter refers to a vertical vision of the paradigm across the layers of the protocol stack and on the performance-oriented integration of the functionalities at different layers.

Indeed, besides the existence of actual cloud computing applications, still they don't address issues related to the underlying transport infrastructure and how to interact and collaborate with it - focusing on universal access rather than performance or predictability.

The next sections provide a brief description of the main issues associated with each challenge.

Assuring the Target QoS/QoE

With the aim of engineering the network so that the target QoS and QoE (Quality of Experience) requirements in a distributed architecture are met, the DS-TE (DiffServ aware Traffic Engineering) architecture represents one of most appropriate solutions for the transport of the relevant Cloud Computing application traffic flows. It manages the QoS in a scalable, flexible and dynamic way and allows for performing Traffic Engineering (TE) in a differentiated service environment by applying routing constrains with class granularity [19]. In a DS-TE domain, TE policies are performed in a per-class basis through DiffServ classification. This goal is achieved by introducing three new concepts:

- Class Type (CT): is a set of traffic trunks with the same QoS requirements. In a DS-TE domain, no more than eight CTs can be set up, on the basis of traffic trunks CoS (Class of Services) values.
- TE-Class: is a combination of a CT and a preemption value, defined as <CT, p>. It allows traffic trunks belonging to the same CT to be forwarded over different LSPs (Label Switched Path) at different priorities. Preemption allows high-priority LSPs to be routed through paths in use by low-priority LSPs, which are then terminated or re-routed. TE-Classes are defined by assigning one or more preemption values to each CT. The maximum number of TE-Class is eight and the belonging of a packet to a TE-Class arises

from the EXP bits, which is a field in the MPLS header.

- Bandwidth Constraint (BC) model: specifies the amount of links' bandwidth that can be used to route LSP belonging to each CT. To this, there are appropriate Bandwidth Constraints defined for each CT and each link.

Regardless of which BC model is adopted, the resulting network performance and resource utilization depend on both the aggregation of the CoS in the implemented CTs and the setting of the BCs. The bandwidth constraints over all the network links have a direct impact on the performance of the constrained-based routing [15], which then heavily influences the call block probability and resulting QoS. It also affects the frequencies of the preemption occurrences, which have to be kept as low as possible [29]. BCs setting together with the selected aggregation of the traffic into the active CTs are also major tasks to control the end-to-end performance in term of delay, losses and jitter. In fact, the belonging of a traffic flow to a certain CT determines the priority of the relevant packets with respect to the others. Additionally, the amount of packets with higher priority depends on the BCs set for the high priority classes. For this reason the setting of the BC and aggregation of the CoS in CTs are problems that have to be jointly addressed. Indeed, traffic flows with dissimilar features characterize the Cloud Computing scenario, such as: bulk data transfers between data-storage centers, short control messages between distributed applications, constant real-time flows for multimedia communications.

Herein, we present a generic DS-TE bandwidth management procedure, which aims to configure the BC Model in terms of the effective network requirements when providing Cloud Computing services. Accordingly, the system adopts the solution of implementing a single algorithm to achieve both Class Type classification and BC definition. This approach performs the two tasks in an interdependent way, optimizing the output of the one in terms of the solution obtained for the other and vice versa.

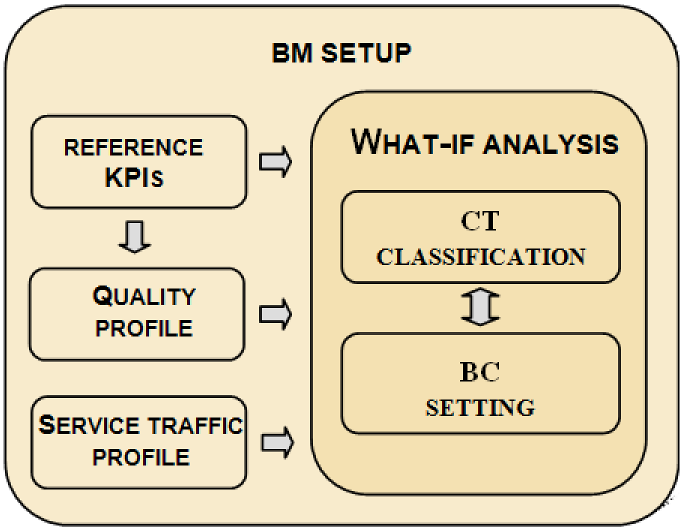

Figure 3: Setup of bandwidth management.

As shown in Figure 3, the bandwidth management setup works on the basis of the following input information: a reference set of key performance indicators (KPIs), which allow for characterizing the service requirements and evaluating QoS network performance; a quality profile, which defines the services classification into CoSs; and the profile of the forecasted ingress traffic. This information, together with the BC model adopted in the network, are the input to a "what-if analysis" to examine network performance and resource utilization at varying CT classifications and BC settings. Note that while the setting of the BC model may vary from link to link, CT classification has to be unique for the whole cloud network.

1. Key Performance Indicators (KPIs)

The Key Performance Indicators provide a quantitative and objective solution to compare the obtained quality performance with the desired ones and are used for both traffic classification and network analysis. Quality can be evaluated from the point of view of the end-user, as perceived quality which refers to the experience in the use of a service, or from the point of view of the operators, as offered quality which refers to the policies of service provisioning, the sustained costs and the capacity to maintain service availability. It can even be intrinsic if it refers to technical network performance parameters, such as delay, jitter, packet loss, and throughput. It can be specified either quantitatively or qualitatively.

Several QoS definitions have been proposed in the literature [13, 34]. However, the followings are those that are most often selected: IP Packet Transfer Delay (IPTD); IP Packet Delay Variation (IPDV); IP Packet Loss Ratio (IPLR); IP Packet Error Ratio (IPER); and Spurious IP Packet Rate (SIPR).

2. Quality profile

The definition of the quality profile consists in identifying the QoS requirements and performing DiffServ classification for the set of services {S} required in the cloud computing scenario. Service characterization and classification rely on a set of KPIs. Let *P* be the number of selected reference KPIs; the *i*-th service within {S} is associated to a vector that specifies its quality requirements:

$$< S >_i = [\Delta KPI_{1,i}^{s}, \Delta KPI_{2,i}^{s}, \Delta KPI_{3,i}^{s}, \ldots, \Delta KPI_{P,i}^{s}].$$

Each element of this vector defines the threshold for the allowed values for each KPI. On the basis of these values, each service is classified into a specific CoS according to the DiffServ model. The 14 standard CoSs [15] can be adopted or new CoSs can be defined by the network operator. One or more services can be classified in the same CoS. As a result of this procedure another vector of KPI is obtained:

$$< CoS >_j = [\Delta KPI_{1,j}^{c}, \Delta KPI_{2,j}^{c}, \Delta KPIC_{3,j}^{c}, \ldots, \Delta KPI_{P,j}^{c}].$$

This vector containing the KPI threshold values which the *j*-th CoS is able to satisfy. They correspond to the intersection of quality requirements of all services in the *j*-th CoS. The cardinality of the set {CoS} can be lower than the number of services to be provided.

3. Service traffic profile

In accordance with the IETF rules, the BC model specifications have to be defined link-by-link; as a consequence, the proposed DS-TE bandwidth management procedure needs to perform a traffic prediction for each link in the network. This prediction can be obtained through a network analysis by considering the following inputs which are available to the operator:

- cloud network topology;
- bandwidth required by each cloud computing service, B_i^s ;
- number of users $U_i^{a,b}$ of service *i* accessing the network at node a and ending at node *b*.

To estimate the traffic load per link, it is necessary to consider the paths between each pair of ingress-egress routers. For the generic edge routers (*a*,*b*), where *a* and *b* are the ingress and egress nodes, respectively, the load for service *i* is equal to:

$$C_i^{a,b} = B_i^s \times U_i^{a,b} \qquad (1)$$

This traffic spreads over the set of available paths from *a* to *b*. The distribution of the traffic through the available paths is evaluated through an empirical simple algorithm, which distributes the traffic according to the path length. In particular, an inverse linear relationship between the traffic load and the path length can be adopted. Only the disjoint *Z* paths no longer than two times the shortest one are considered in this process and the traffic is distributed according to the length of each path p_z. The traffic load from *a* to *b* along the *z*-path can be assumed to be equal to:

$$c_{i,z}^{a,b} = C_i^{a,b} \frac{1}{p_z \sum_{x=1}^{Z} 1/p_x} \qquad (2)$$

From this distribution, the total load per link and per service is computed.

Note that the proposed algorithm is quite simple. The choice has been driven by the fact that at this stage it is just required to find enough resources from the source to the destination to satisfy the bandwidth demands for the different services. For this purpose, the use of complex routing procedures would be useless [15]. These are instead adopted when addressing LSP setup requests.

At this stage the maximum length among all the paths traversing each link needs to be computed. This is used to compute local KPI thresholds from the end-to-end KPI thresholds in <CoS>j, as discussed in the following.

4. What-if analysis

This procedure is intended to provide a solution to the problem of mapping the cloud computing services into the appropriate Class Type and to found out the optimal setting of the bandwidth constraints. Herein, appropriateness and optimally are defined according to the KPI constraints (for each service) and in terms of resource utilization and/or saving.

The first step is the detection of a possible CT classification, which is performed by evaluating the <CoS> KPI vectors which were defined during the quality profile definition phase. The possible mappings from CoSs to CTs can be obtained in two possible ways:

- activating a CT for each CoS (since the maximum number of activable CTs is 8, this solution is possible only if the number of CoSs is lower than 8).
- grouping more CoSs in the same CT. In this case, the bandwidth allocation benefits from reducing the number of CTs at the expense of a lower efficiency in terms of QoS

requirements satisfaction. The allowed combinations of CoSs are those which satisfy the following conditions:

- at least three CTs are defined: CT2 for expedited traffic, CT1 with intermediate guarantees and CT0 for best effort services;
- the priority order defined by DiffServ classification is respected (only consecutive CoSs are grouped).

If W is the cardinality of the set {CoS}, the resulting total number of {CT} classifications is:

$$H = \sum_{v=3}^{V} \binom{W-1}{W-v} \qquad (3)$$

where $V = W$ if $W \leq 8$ and $V = 8$ otherwise.

Each k-th CT of the considered classification needs to satisfy the quality parameters of the encompassed CoSs so that another vector of KPIs <CT> is defined as the intersection of all corresponding <CoS>$_j$. These represent the KPI thresholds for all the services included in each CT.

Once all the possible aggregations have been defined, these are evaluated by computing a gain function that takes into account both KPI gain (GKPI) and bandwidth gain (GBDW). GKPI evaluates to which extent the considered solution allows for keeping each KPIs lower than the desired thresholds <CoS>$_j$. GBDW provides a measure of the amount of bandwidth that is still free for each CT when applying the proposed solution. While the meaning and the objectives of both these functions are clearly defined, the exact formula to be used depends on the specific scenario under investigation.

This analysis is performed for all other {CT}h until $h = H$. Then, the optimal CT classification and the optimal BCs for each link are identified by selecting the solution with the highest gain function among all the evaluated combinations.

Service Characterization and Traffic Modeling

A suitable framework for performance-oriented analysis of cloud computing architectures and solutions is needed, which in turn implies the introduction of an effective characterization of the services and their footprint on the transport infrastructure, and in general the definition of service requirements and their dependence on the number of users.

In general, an open issue is related to the quantification of a service, since especially in the case of cloud computing a service represents a multi-dimensional load on the computing and networking infrastructure – both in terms of computational load and of network load (including bandwidth, requirements, etc.). The idea here proposed is to instrument cloud computing platforms with traffic classification features useful to both cloud traffic and applications classification. This permits to clear understand (i) the traffic received/generated by the cloud; (ii) the applications responsible for a specific portion of traffic (e.g., heavy hitters); (iii) the relationships between traffic flows and cloud applications.

Therefore, there is a strong need for new methodologies and techniques for service characterization and for understanding the impact of cloud computing services on the network. We propose a two step approach:

- the first step of this analysis is the study of the interactions among the entities of the considered distributed applications, that is service characterization. This task provides insights related to the process of the composition and use of a service. This step will study the cloud computing services and their characterization from the following viewpoints: (i) interactions among the several entities of the considered service; (ii) interactions between users and services; (iii) interactions between users and networks.

- The second step is the characterization of the traffic generated by cloud computing applications: after the "high-level" analysis of the previous step, in this second step the target of the traffic analysis and footprinting stage is the understanding of the dynamics associated to cloud computing services and final objectives are: (i) to gain knowledge on the traffic generated by these services; (ii) to study techniques and methodologies for the identification and classification of the traffic generated by cloud computing services; (iii) to improve the support for such new services; (iv) to guarantee their reliability and proper use; (v) to better plan their evolution; (vi) to develop new network architectures for better supporting them.

Signaling

Introducing specific signaling structures and protocols is required to enable tighter cooperation between cloud computing applications and the underlying transport infrastructure. In this framework, the cloud architecture could benefit from the definition of suitable built-in monitoring primitives aimed at providing information about the number of users, SLAs, network context and resources.

In the following, possible cross-layer signaling architectures are briefly described:

- *Interlayer signaling pipe* allows propagation of signaling messages layer-to-layer along with packet data flow inside the protocol stack in bottom-up or top-down manner. Signaling information propagates along with the data flow inside the protocol stack and can be associated with a particular packet incoming or outgoing from the protocol stack. Packet headers or packet structures are two techniques considered for encapsulation of signaling information and its propagation along the protocol stack.

- *Direct Interlayer Communication* introduces signaling “shortcuts” performed out of band allowing non-neighboring layers of the protocol stack to exchange messages, without processing at every adjacent layer, thus allowing fast signaling information delivery to the destination layer. Despite the advantages of direct communication between protocol layers this approach is mostly limited by request-response action - while more complicated event-based signaling should be adapted. To this aim, a mechanism which uses callback functions can be employed. This mechanism allows a given protocol layer to register a specific procedure (callback function) with another protocol layer, whose execution is triggered by a specific event at that layer.

- *Central Cross-layer Plane* implemented in parallel to the protocol stack is the most widely proposed cross-layer signaling architecture. Typically, it is implemented using a shared bus or database that can be accessed by all layers. Parameter exchanged between layers is standardized and performed using well-defined layer interfacing modules each of which exports a set of IP functions.
- *Network-wide Cross-Layer Signaling* represents a novel approach, allowing network-wide propagation of cross-layer signaling information adding another degree of freedom in how cross-layer signaling can be performed. Implementation of network-wide cross-layering should include a combination of signaling techniques (like packet headers, standalone messages, or feedback functions) depending on signaling goals and the scope (at the node or in the network) the cross-layer signaling is performed.

The choice of a specific signaling architecture should be driven by quantitative analysis in terms of communication requirements, resulting overhead, interoperability, etc. A specific framework should be identified to address such issue and provide a design tool for cloud computing system engineers.

Overlay and Performance Optimization

As dynamic adaptation and optimization will represent the central paradigm of a cloud computing platform, the issue represents a core problem. Optimization of cloud computing systems, with specific emphasis on supporting effective scalability, dependability, and reliability of the solutions under a dynamic perspective, involves several interdisciplinary areas.

As all distributed applications, cloud computing involves the definition of a virtual topology or "overlay" in terms of logical connections among the servers, services, clients and other entities in the application domain. The virtual topology is then mapped onto the network topology, which is the actual communication infrastructure. A specific issue which raises in this process is whether such mapping operation between the two topologies should be blind or resource-aware. Similar issues, even if related to computational time optimization, were faced in grid computing platform – leading to the definition of specific functionalities such as the Network Weather Service [26]. Clearly, efficient mapping implies suitable signaling architectures (see previous section).

In general, performance optimization in a cloud computing scenario represents a multi-dimensional problem, depending from several variables (network context, operating procedures) and aimed at different but joint optimization goals (response time, execution time, interactivity, etc.). As a consequence, relevant works in the framework of scientific literature on Multi-Objective Optimization (MOO) should be considered and customized and adapted to the cloud computing scenario.

Finally, efficient runtime resource utilization at computational and network levels puts a strong emphasis on the aspect of measurements in order to provide the necessary knowledge about the operating context. In this scenario, measurement-based approaches should be considered in order to support the theoretical benefits deriving from the chosen MOO strategies. Again, relevant works are available in the literature, but not specifically tailored for cloud computing platforms.

Interoperability and Testing

Current architectures for cloud computing offer limited possibilities to the user such as flat prices for resources leasing, proprietary interfaces for services access and the lack of portability for the applications, making very difficult for a user to migrate applications from a service platform to another. In the future, it will be very useful to have a standard for interoperability among the different cloud architectures. This would allow the growth of new applications and the development of a market based approach to cloud computing. This can be studied thanks to distributed experimental facilities and testbeds based on the concept of federation. It is being widely used for network testbeds. In the framework of PlanetLab [32], a peer-to-peer federation between PlanetLab Central (See Figure 4) and PlanetLab Europe (see Figure 5) has been successfully established thanks to the ONELAB European Project [27]. Just to provide the reader a quantitative dimension of these testbeds, PlanetLab currently consists of 1132 nodes at 517 sites (as October 2010).

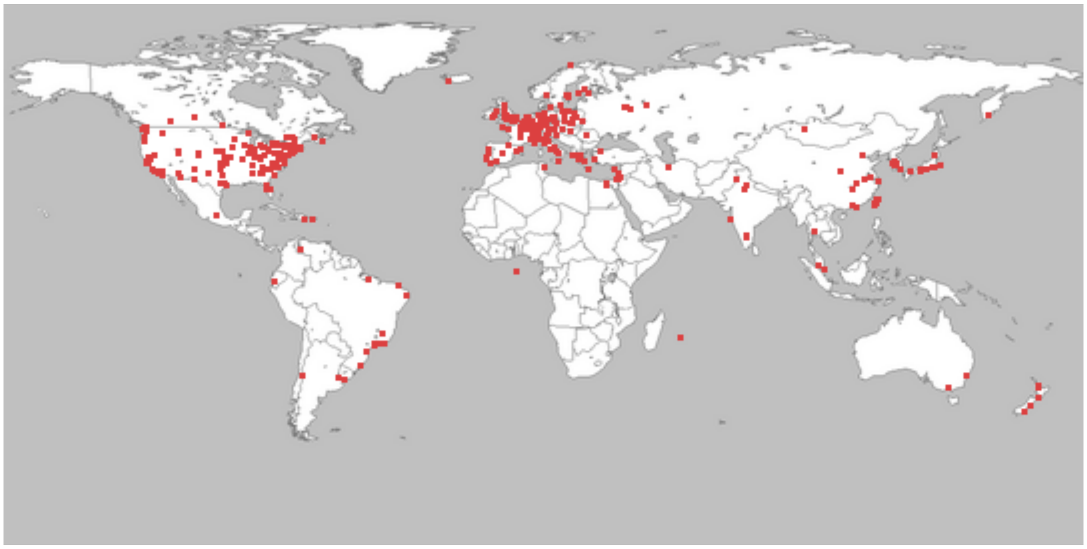

Figure 4: PlanetLab nodes on the Map (Source https://www.planet-lab.org/ - Copyright © 2007 The Trustees of Princeton University).

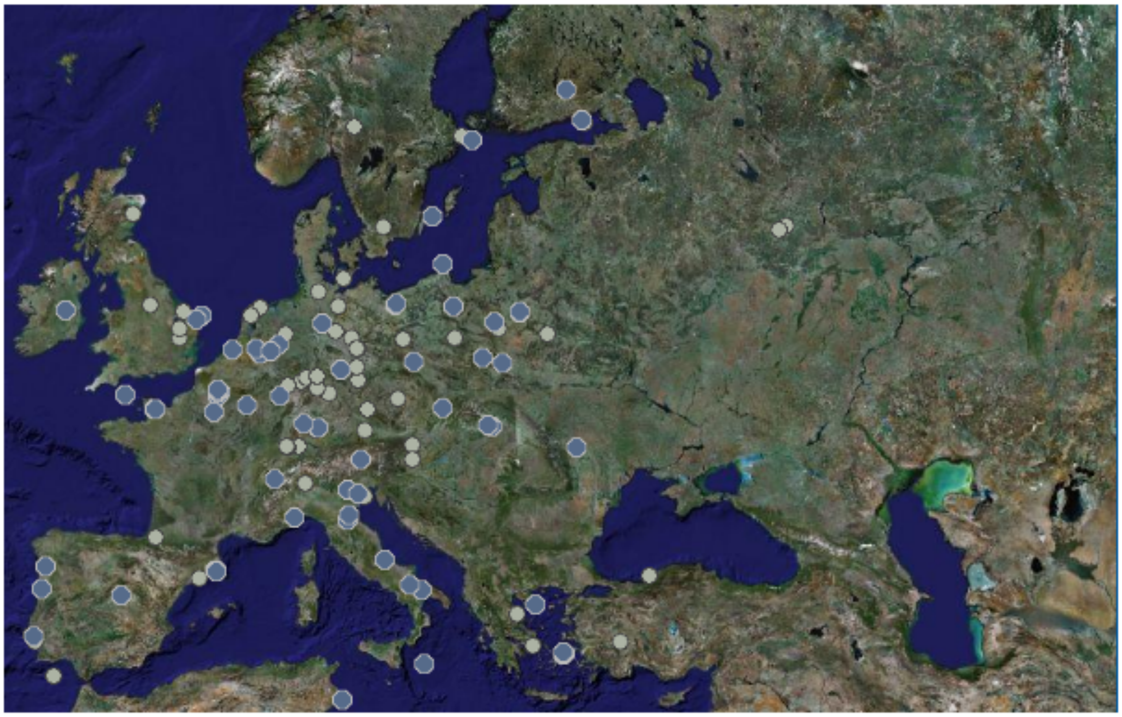

Figure 5: PlanetLab Europe nodes on the Map (Source: http://www.planet-lab.eu/).

DETER [6] testbed is built by implementing a sort of federation between several independent EMULAB-based testbeds [7]. Likewise, a similar federation effort is being developed with the objective of running experiments across different ORBIT [28] wireless testbeds. A more challenging step in this process consists in further extending the concept of federation across heterogeneous testbeds. Federation between PlanetLab and EMULAB is currently investigated. Federation is also addressed by the ONELAB2 project (part of the FIRE initiative [10]), a follow-up of ONELAB started in September 2008.

Future architectures for cloud computing should offer the possibilities to adopt federation approaches to integrate distributed testbeds in a unique experimental facility defining a framework for the interoperability among architectures.

IV. Conclusions and Final Remarks

Cloud computing represents an emerging paradigm in the world of distributed computing. Nevertheless, it provides relevant challenges in the areas of networking and system engineering in general. The chapter proposed a brief overview of available platforms for cloud computing, focusing on current and perspective networking issues to be considered by network engineers in order to control and optimize performance of such computing service. Some of those are already considered in the literature, even if in different domains, while others remain completely uncovered – and therefore represent problems where competences related to networking enter the scenario of cloud computing.

References

[1] L. Atzori, T. Onali, "Operators Challenges towards Bandwidth Management in DiffServ-aware Traffic Engineering Networks", IEEE Comm. Magazine, Vol. 46, n. 5, May 2008.

[2] R. Buyya, C.S. Yeo, S. Venugopal, "Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Facilities," 10th IEEE International Conference on High Performance Computing and Communications, 2008, pp. 5-13.

[3] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, A. Warfield, "Xen and the art of virtualization", ACM Symposium on Operating Systems Principles, SOSP 2003.

[4] Y. Cheng, A. Leon-Garcia, I. Foster, "Toward an Autonomic Service Management Framework: A Holistic Vision of SOA, AON, and Autonomic Computing", IEEE Comm. Magazine, Vol. 46, no. 5, May 2008.

[5] Chappell, David (August 2008). "A Short Introduction to Cloud Platforms". David Chappell & Associates. Retrieved on 2008-08-20.

[6] http://www.isi.edu/deter/

[7] http://www.emulab.net/

[8] I. Foster, Y. Zhao, I. Raicu, S. Lu, "Cloud computing and Grid Computing 360-Degree Compared," 2008 Grid Computing Environments Workshop, Nov. 12-18, 2008, pp. 1-10.

[9] http://www.freebsd.org/doc/en/books/handbook/jails.html

[10] http://cordis.europa.eu/fp7/ict/fire/

[11] Google App Engine, http://appengine.google.com

[12] Liu, Y.; Zhuang, M.; Wang, Q.; Zhang, G., "A New Approach to Web Services Characterization," Asia-Pacific Services Computing Conference, 2008. APSCC '08. IEEE , vol., no., pp.404-409, 9-12 Dec. 2008

[13] ITU-T Rec. G.1540, “IP Packet transfer and Availability Performance Parameters,” Dec. 2002.

[14] ITU-T Rec M.3060/Y.2401, "Principles for the Management of the Next Generation Networks," Mar. 2006.

[15] A. Karaman, “Constraint-Based Routing in Traffic Engineering”, IEEE International Symposium on Computer Networks (ISCN'06), page(s): 1- 6, 16-18 June 2006.

[16] S. Androulidakis, T. Doukoglou, G. Patikis, D. Kagklis, "Service Differentiation and Traffic Engineering in IP over WDM Networks" IEEE Comm. Magazine, vol. 46, n. 5, May 2008.

[17] D.Kliazovich, M.Devetsikiotis, F.Granelli, "Formal Methods in Cross layer Modeling and Optimization of Wireless Networks: State of the Art and Future Directions", in Heterogeneous Next Generation Networking: Innovations and Platform, IGI, 2008.

[18] T. Karagiannis, K. Papagiannaki, M. Faloutsos. Blinc: Multilevel traffic classification in the dark. ACM SIGCOMM, 2005.

[19] Le Faucheur, F., Ed., “Protocol Extensions for Support of Diffserv-aware MPLS Traffic Engineering”, RFC 4124, June 2005.

[20] L. WANG, G. VON LASZEWSKI. "Cloud Computing: a Perspective Study", International Workshop on Grid Computing Environments--Open Access (2008).

[21] L7-filter, Application Layer Packet Classifier for Linux. http://l7-filter.sourceforge.net.

[22] http://linux-vserver.org

[23] Y. Liu; M. Zhuang; B. Yu; G. Zhang; X. Meng, "Services Characterization with Statistical Study on Existing Web Services," Web Services, 2008. ICWS '08. IEEE International Conference on , vol., no., pp.803-804, 23-26 Sept. 2008

[24] T. Auld, A. W. Moore, S. F. Gull. Bayesian neural networks for internet traffic classification. IEEE Transactions on Neural Networks, 2007.

[25] A. Moore, D. Zuev. Internet traffic classification using bayesian analysis techniques. ACM SIGMETRICS, June 2005.

[26] “Network Weather Service,” http://nws.cs.ucsb.edu

[27] http://www.onelab.eu/

[28] http://www.orbit-lab.org/

[29] J.C. de Oliveira, C. Scoglio, I.F. Akyildiz, G. Uhl, “New Preemption Policies for DiffServ-Aware Traffic Engineering to Minimize Rerouting in MPLS Networks,” IEEE/ACM Trans. on Networking, vol. 12, no. 4, August 2004.

[30] http://openvz.org

[31] G. Popek and R. Goldberg. Formal requirements for virtualizable third generation architectures. Commum. ACM, 17(7):412-421, 1974.

[32] http://www.planet-lab.org/

[33] P. Ruth, X. Jiang, D. Xu, S. Goasguen, "Virtual Distributed Environments in a Shared Infrastructure", IEEE Computer, Special Issue on Virtualization Technologies, May 2005.

[34] N. Seitz, “ITU-T QoS standards for IP-based networks,” IEEE Communication Magazine, June 2003.

[35] L. Bernaille, R. Teixeira, K. Salamatian. Early application Identification. ACM CoNEXT, December 2006.

[36] H. Wang, G. Wang, C. Wang, A. Chen, R. Santiago, "Service Level Management in Global Enterprise Services: from QoS Monitoring and Diagnostics to Adaptation, a Case Study", IEEE ECC, 2007.

[37] M. McNett, D. Gupta, A. Vahdat, G. M. Voelker, "Usher: An Extensible Framework for Managing Clusters of Virtual Machines", UENIX Large Installation System Administration Conference (LISA), 2007.

[38] R. Buyya; C. S. Yeo, S. Venugopal. "Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities", IEEE International Conference on High Performance Computing and Communications, 2008. Melbourne, Australia, 25-27 Sept. 2008

[39] V. Paxson. Bro: A system for detecting network intruders in realtime. Computer Networks, 1999.

[40] Yahoo Pipes, http://pipes.yahoo.com/pipes/

[41] Breiter, G.; Behrendt, M.; , "Life cycle and characteristics of services in the world of cloud computing," IBM Journal of Research and Development , vol.53, no.4, pp.3:1-3:8, July 2009

[42] M.D. Dikaiakos, D. Katsaros, P. Mehra, G. Pallis, A. Vakali, "Cloud Computing: Distributed Internet Computing for IT and Scientific Research," Internet Computing, IEEE , vol.13, no.5, pp.10-13, Sept.-Oct. 2009

[43] Wei, Yi; Blake, M. Brian, "Service-Oriented Computing and Cloud Computing: Challenges and Opportunities," Internet Computing, IEEE , vol.14, no.6, pp.72-75, Nov.-Dec. 2010

[44] G. Pallis, G, "Cloud Computing: The New Frontier of Internet Computing," Internet Computing, IEEE , vol.14, no.5, pp.70-73, Sept.-Oct. 2010