A Survey on Open-source Cloud Computing Solutions

Patrícia Takako Endo[1], Glauco Estácio Gonçalves[1], Judith Kelner[1], Djamel Sadok[1]

[1]Universidade Federal de Pernambuco – UFPE
Caixa Postal 7851 – 50732-970 – Recife – PE

{patricia,glauco,jk,jamel}@gprt.ufpe.br

Abstract. *Cloud computing is an attractive computing model since it allows for resources to be provisioned according on a demand basis, i.e., cloud users can rent resources as they become necessary. This model motivated several academic and non-academic institutions to develop open-source cloud solutions. This paper presents and discusses the state-of-the of open-source solutions for cloud computing. The authors hope that the observation and classification of such solutions can leverage the cloud computing research area providing a good starting point to cope with some of the problems present in cloud computing environments.*

1. Introduction

At present, it is common to access content across the Internet with little reference to the underlying hosting infrastructure, which is formed by data centers maintained by content providers. The entire technology used to provide such level of locality transparency offers also a new model for the provision of computing services, stated as Cloud Computing. In such a model, enterprises put their applications in the cloud – a very large computational entity from a developer´s perspective – without concern for where the services are actually hosted or how they are delivered. Through access to slice of computational power over a scalable network of nodes, enterprises can reduce or eliminate costs associated with internal infrastructure for the provision of their services.

A further viewpoint of cloud users, in addition to costs reduction, comes from the fact that the cloud computing model can be attractive since it allows resources to be provisioned according to the enterprise needs. This is in contrast, to traditional practices, where resources are often dimensioned according to the worst case use and peak scenarios. Hence cloud users can rent resources as they become necessary, in a much more scalable and elastic way. Moreover, enterprises can transfer operational risks to cloud providers. In the viewpoint of cloud providers, the model offers a way for better utilization of their own infrastructure. Authors in [Armbrust et al 2009] point that this model benefits from a form of statistical multiplexing, since it allocates resources for several users concurrently. This statistical multiplexing of data centers is guaranteed through several decades of research in many areas as: distributed computing, grid computing, web technologies, service computing, and virtualization. Several authors ([Armbrust et al 2009], [Vaquero et al 2008], [Buyya et al 2009]) agree that from such research areas virtualization brought the key technologies to leverage cloud computing.

Despite the increasingly widespread use of the Cloud Computing term, there is no formal definition for it yet. In a recent paper [Vaquero et al 2008], authors review the

cloud literature for a minimum set of characteristics that cloud solutions must present. But they were not able to find even a single common feature in literature definitions. They note that the set of characteristics that at most are similar to a minimum common definition are: scalability, pay-per use model, and virtualization. Finally, using these features the authors gave their own definition: “Clouds are a large pool of easily usable and accessible virtualized resources. These resources can be dynamically reconfigured to adjust to a variable load, allowing also for an optimum resource utilization. This pool of resources is typically exploited by a pay-per-use model in which guarantees are offered by the Infrastructure Provider by means of customized SLA”.

Since the popularization of the cloud computing term in 2007, with IBM Blue Cloud [Vouk 2008], several enterprises become cloud computing providers: Amazon and their Elastic Compute Cloud (EC2)[1], Google with Google App Engine[2], Microsoft with their Windows Azure Platform[3], Salesforce and their Force.com[4] and so on. Though these solutions fit the cloud computing definition they differ in their programmability. A concept borrowed from the network virtualization area [Chowdhury and Boutaba 2010], programmability is relative to the programming features a network element offers to developers, measuring how much freedom the developer has to manipulate resources and/or devices.

This concept can be applied to compare cloud computing solutions. More programmable clouds offer environments where developers are free to choose their own programming paradigm, languages, and platforms, having total control over their virtualized resources. Less programmable clouds restrict developers in some way: forcing a set of programming languages, or allowing support for only one application paradigm. On the other hand, a high level of programmability turns hard the cloud management because the cloud provider will have a much more heterogeneous environment to manage. For example, if a cloud provider allows their users to use any operating system in a virtual machine, the cloud operator will have to cope with a large number of solutions to provide fault-tolerance services. Moreover, less programmable solutions abstract some operational issues (processes communication, network access, storage functions, etc.) through some type of middleware. An instance of a cloud solution with high programmability is the Amazon EC2, where users can choose their operating system from a set of supported operating systems given by Amazon and they can configure their virtual machines to work as they see fit. The Google App Engine is an example of a less programmable solution, since it restricts developers to the Web paradigm and to some programming languages.

A common characteristic present in all the cited cloud solutions is that cloud owners understandably avoid revealing the underlying features of their solution, since this is seen as part of their strategic information. Despite this, one may point out the considerable efforts made by several academic and non-academic institutions to develop open-source cloud solutions. The authors hope that the observation and classification of

[1] http://aws.amazon.com/ec2/

[2] http://code.google.com/intl/appengine/appengine/

[3] http://www.microsoft.com/windowsazure/

[4] http://www.salesforce.com/platform/

such solutions can leverage further the cloud computing research area providing a good starting point to discovering different approaches to the problems present in cloud computing environments.

This paper presents and discusses the state-of-the of open-source solutions for cloud computing. The remainder of this paper is organized as follows. Next section introduces the main challenges developers face to create cloud computing solutions. Section III shows efforts in standardizing cloud computing interfaces for interoperability. Section IV introduces the main open-source cloud architectures and Section V makes comparisons between the architectures. Finally, a conclusion of this paper and future works are presented in Sections VI.

2. Challenges

The development of cloud computing solutions brings several technical challenges to cloud developers. These challenges can be grouped in three main areas: negotiation, decision, and operation. In the negotiation area, these are the challenges relative to how application developers interface with the cloud as well as the description of the cloud offerings. It includes also the definition of the programmability level that the cloud solution will offer. The decision area copes with the main problem that clouds faces behind the scenes: How virtual resources can be scheduled to meet user requirements, for example? Last, the operation area is associated with the enforcement of decisions and the communication between cloud elements. The following sub-sections discuss in details the main challenges in each one of these areas.

2.1. Negotiation

The negotiation area concerns itself with challenges relative to the interface between the application developers and the cloud. Generally, the interface between the cloud and application developers assumes the form of an Application Programming Interface (API), but, depending on the programmability level offered by the cloud, this API can be implemented in several ways ranging from a web-service based toolkit to control virtual machines in the cloud to a set of programming primitives used to develop distributed applications in the cloud. In addition to these basic functions, such APIs must allow developers to request – and control, possibly – additional functionalities offered by a cloud operator like service quality assessment, load balance, elastic application growth, backup strategies, and so on. There are still some other requirements that can be considered in the cloud API like geographical restrictions enforced to allocate virtual machines due to legal issues. One may think of some type of content or application that is strategically limited to a country or a region for copyright or security reasons.

At the present, APIs and the negotiation process offered by cloud providers follow a semi-automatic scheme where a human interacts with the cloud through programming primitives or a visual interface. But, next generation clouds can offer sophisticated ways to interact with human users through high-level abstractions and service-level policies. Such an interface type will need some formalism to specify both cloud offerings and application requirements as well as offering the support for an automatic negotiation process.

2.2. Decision

The main target of any cloud operator is to schedule developer applications aiming for the maximum utilization of cloud resources. A developer´s application covers, beyond the actual code, some additional information about application´s needs and services negotiated previously. In other words, one can abstract this additional information to some type of network virtualization demand with a topology formed by virtual nodes where the application runs and virtual links for communication. Thus, the cloud operator problem turns into that of selecting the best suitable physical resources to accommodate these virtual resources.

This careful mapping requires advanced strategies. The first challenge imposed by this optimization problem is that it is NP-hard [Andersen 2002] and hence any useful solution would need to relax some of its problem conditions and constraints to obtain an approximate solution in a polynomial time. The second challenge is to meet all the clauses negotiated with the developer. Depending on the nature of such contract, application scheduling algorithms will cope with quality restrictions, jurisdiction restrictions, elastic growth, and so on.

2.3. Operation

Metaphorically, one can say that while in the decision area the cloud operator must identify solutions for the “brain” of the cloud, in the operation area it must attack the problems of the “limbs” of the cloud, i.e., they must provide some form to enforce decisions. The enforcement here covers the communication protocols and the configuration of cloud elements.

A communication protocol can be used to monitor and reserve resources in the cloud. The cloud is composed by different elements like processing servers, switches, routers, links and storage components. Due to such heterogeneity, the communication between the decision-maker and elements puts a challenge on cloud design. Overall, existing cloud solutions use Web Services to provide communication with processing and storage nodes, but many communication elements do not support such implementations. Thus, cloud architects are using traditional traffic engineering protocols to provide reservation of network elements. One possible idea to cope with this challenge is to use smart communication nodes with an open interface to create new services in the node the emerging Openflow-enabled switches [McKeown et al 2008].

Node communication is just one part of the problem; the other one is to configure this. Here, the recent advances in server virtualization have provided several solutions for operators to benefit from.

3. Standardization efforts

A considerable challenge present in many of the raised discussions around the cloud is related to the need for standardization. All the three areas presented in Section 2 face the standardization challenge in some way, but the main challenge occurs in the negotiation area. Currently, cloud providers offer proprietary interfaces to access their services. This locks users within a given provider as they cannot migrate their applications and services easily between cloud providers [Buyya et al 2009]. It is hoped that cloud

providers see such a problem and work together to offer a standardized API based on open standards like SOAP and REST.

An important effort in the standardization comes from the Open Cloud Manifesto [OpenCloud 2009]. This is an initiative supported by hundreds of companies that aims to discuss with cloud organizations a way to produce open standards for cloud computing. Their major doctrines are collaboration and coordination of efforts on the standardization, adoption of open standards wherever appropriate, and the development of standards based on customer requirements. Participants of the Open Cloud Manifesto through the Cloud Computing Use Case group produced an interesting white paper [OpenCloud 2010] highlighting the requirements that need to be standardized in a cloud environment to ensure interoperability in the most typical scenarios of interaction – Use Cases – in cloud computing.

4. Open-source solutions for Cloud Computing

Due to the large growth of cloud computing, there are several solutions in this area. This article is focused on open source solutions, highlighting their main characteristics and architectures proposed.

4.1. Xen Cloud Platform (XCP)

The Xen hypervisor [Citrix Systems 2010b] is a solution for infrastructure virtualization that provides an abstraction layer between servers' hardware and the operating system. A Xen hypervisor allows each physical server to run several "virtual servers" handling the operating system and its applications from the underlying physical server. The Xen solution is used by many cloud solutions such as Amazon EC2, Nimbus and Eucalyptus.

Recently, Xen.org announced the Xen Cloud Platform (XCP) [Citrix Systems 2010a] as a solution for cloud infrastructure virtualization. But, differently from existent open source cloud solutions, XCP does not provide the overall architecture for cloud services. Their goal is to provide a tool to cope with automatic configuration and maintenance of cloud platforms [Citrix Systems 2010c].

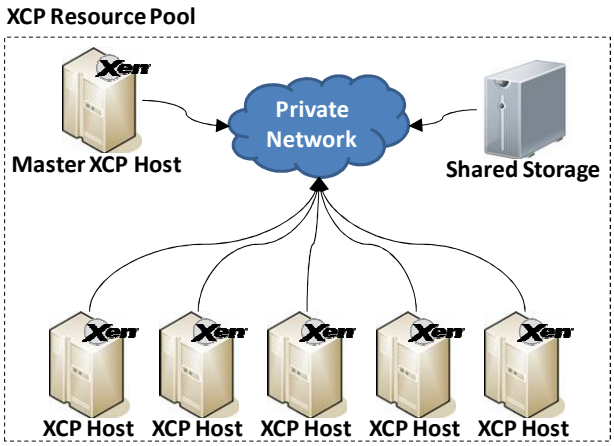

Figure 1. XCP Architecture

The XCP architecture (Figure 1) is based on the **XCP hosts** that are responsible to host the virtual machines. According to [Xen.org. 2009], these hosts are aggregated in

a **XCP resource pool** and using a **Shared Storage** the virtual machines can be started and restarted on any XCP host. The **Master XCP host** offers an administration interface and forwards command messages to others XCP hosts.

4.2. Nimbus

Nimbus [Keahey 2009] is an open source solution (licensed under the terms of the Apache License) to turn clusters into an Infrastructure as a Service (IaaS) for Cloud Computing focusing mainly on scientific applications.

This solution gives to users the possibility to allocate and configure remote resources by deploying VMs – known as Virtual Workspace Service (VWS). A VWS is a VM manager that different frontends can invoke.

To deploy applications, Nimbus offers a "cloudkit" configuration that consists of a manager service hosting and an image repository. The workspace components are shown in Figure 2.

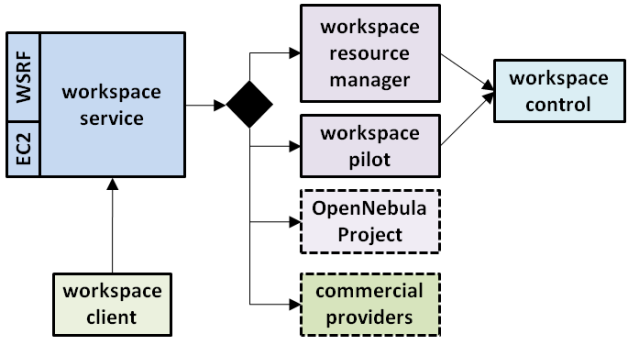

Figure 2. Nimbus workspace components [Keahey 2009]

- **Workspace service**: is web services based and provides security with the GSI authentication and authorization. Currently, Nimbus supports two frontends: Amazon EC2 and WSRF.
- **Workspace control**: is responsible for controlling VM instances, managing and reconstructing images, integrating a VM to the network and assigning IP and MAC addresses. The workspace control tools operate with the Xen hypervisor and can also operate with KVM[5].
- **Workspace resource management**: is an open source solution to manage different VMs, but can be replaced by other technologies such as OpenNebula.
- **Workspace pilot**: is responsible for providing virtualization with few changes in cluster operation. This component handles signals and has administration tools.

4.3. OpenNebula

OpenNebula [OpenNebula Project 2010] is an open-source toolkit used to build private, public and hybrid clouds. It has been designed to be integrated with networking and storage solutions and to fit into existing data centers.

[5] http://www.linux-kvm.org/page/Main_Page

The OpenNebula architecture (Figure 3) is based on three basic technologies to enable the provision of services on a distributed infrastructure: virtualization, storage and network. All resource allocation is done based on policies.

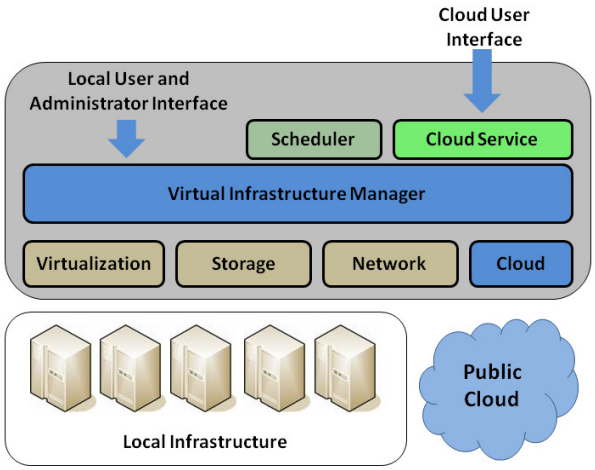

Figure 3. OpenNebula architecture [OpenNebula Project 2010]

The Cumulus Project [Wang et al 2008] is an academic proposal based on OpenNebula. Cumulus intends to provide virtual machines, virtual applications and virtual computing platforms for scientific applications. Visualizing the integration of already existing technologies, the Cumulus project uses HP and IBM blade serves running Linux and Xen hypervisor.

The Cumulus networking solution was called the "*forward*" mode, where users do not need to specify any network configuration information. Instead the backend servers are responsible for allocating a dynamic IP address for a VM and returning these to the users, making such networking solution transparent to the users.

The Cumulus design is a layered architecture (Figure 4) with three main entities: Cumulus frontend, OpenNebula frontend, and OS Farm. This proposal focuses on reaching scalability and autonomy of data centers.

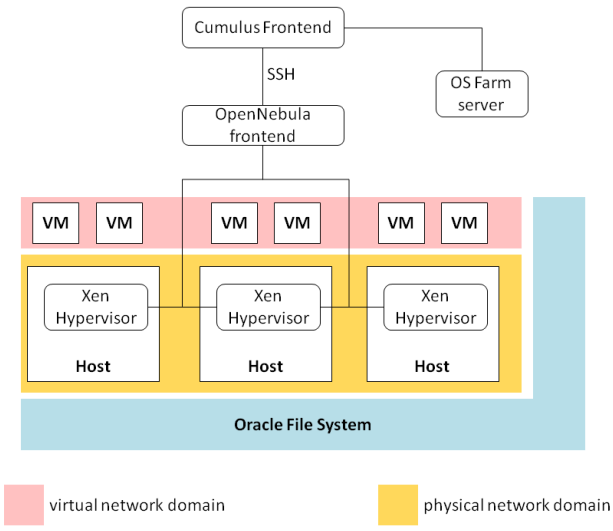

Figure 4. Cumulus architecture [Wang et al 2008]

- **Cumulus frontend**: the Cumulus frontend is the access point for a Cumulus system and is responsible for handling VM requirements.
- **OpenNebula frontend**: the OpenNebula frontend provides an interface to manage the distributed blade servers and the resources for VM deployment. To administrate a common user system, Cumulus uses NIS (Network Information System) and NFS (Network File System) to manage shared directory. Moreover, OpenNebula was merged with secure infrastructure solutions, such as LDAP (Lightweight Directory Access Protocol) and the Oracle Cluster File System.
- **OS Farm**: the OS Farm is a tool for VM template management that operates to generate and to store Xen VM images and virtual appliances.

4.4. Eucalyptus

Eucalyptus [Nurmi et al 2009] is an open source cloud computing framework focused on academic research. It provides resources for experimental instrumentation and study. Eucalyptus users are able to start, control, access and terminate entire virtual machines. In its current version, Eucalyptus supports VMs that run atop the Xen supervisor [Barham et al 2003].

According to [Nurmi et al 2009], the Eucalyptus project presents four characteristics that differentiate it from others cloud computing solutions: a) Eucalyptus was designed to be simple without requiring dedicated resources; b) Eucalyptus was designed to encourage third-party extensions through modular software framework and language-agnostic communication mechanisms; c) Eucalyptus external interface is based on the Amazon API (Amazon EC2) and d) Eucalyptus provides a virtual network overlay that both isolates network traffic of different users and allows clusters to appear to be part of the same local network.

The Eucalyptus architecture is hierarchical (Figure 5) and made up of four high level components, where each one is implemented as a stand-alone web service.

- **Node Controller (NC)**: this component runs on every node that is destined for hosting VM instances. An NC is responsible to query and control the system software (operating system and hypervisor) and for conforming requests from its respective Cluster Controller. The role of NC queries is to collect essential information, such as the node's physical resources (e.g. the number of cores and the available disk space) and the state of VM instances on the nodes. NC sends this information to its Cluster Controller (CC). NC is also responsible for assisting CC to control VM instances on a node, verifying the authorization, confirming resources availability and executing the request with the hypervisor.
- **Cluster Controller (CC)**: this component generally executes on a cluster front-end machine, or any machine that has network connectivity to two nodes: one running NCs and another running the Cloud Controller (CLC). A CC is responsible to collect/report information about and schedule VM execution on specific NCs and to manage virtual instance network overlay.
- **Storage Controller (Walrus)**: this component is a data storage service that provides a mechanism for storing and accessing virtual machine images and user

data. Walrus is based on web services technologies and compatible with Amazon's Simple Storage Service (S3) interface [Amazon 2006].

- **Cloud Controller (CLC)**: this component is the entry-point into the cloud for users. Its main goal is to offer and manage the Eucalyptus underlying virtualized resources. CLC is responsible for querying node managers for resources' information, making scheduling decisions, and implementing them by requests to CC. This component is composed by a set of web services which can be grouped into three categories, according their roles: resource services, data services, and interface services.

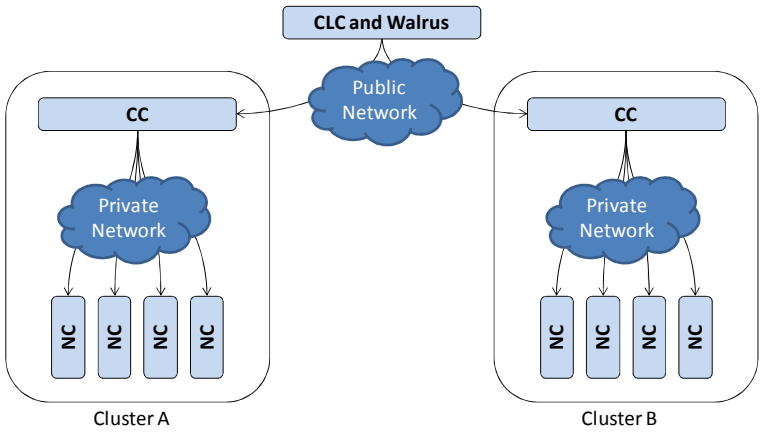

Figure 5. Eucalyptus architecture [Nurmi et al 2009]

Ubuntu Enterprise Cloud (UEC)[6] is an Amazon EC2 like infrastructure and is powered by Eucalyptus. Its main goal is to provide a simple process of building and managing internal infrastructure for cloud. The Ubuntu 9.04 Server Edition is integrated with Eucalyptus that uses the KVM hypervisor.

The UEC architecture is based on the Eucalyptus architecture in which each elements is an independent web service that publishes a Web Service Description Language (WSDL) document defining the API to interact with it.

Furthermore, UEC defines three layers for security: authentication and authorization, network isolation and Machine Instance Isolation (MInst). The authentication and authorization layer is responsible for locally generated X.509 certificates; the network isolation layer is important to prevent eavesdropping of network traffic and; the MInst layer consists of Networking isolation, Operating System isolation, and Hypervisor based machine isolation.

4.5. TPlatform

TPlatform [Peng et al 2009] is a cloud solution that provides a development platform for web mining applications, which is inspired in Google cloud technologies, and which acts as a Platform as a Service (PaaS) solution. Their infrastructure is supported by three technologies: a scalable file system called Tianwang File System (TFS) what is similar to the Google File System (GFS), the BigTable data storage mechanism, and the

[6] http://www.ubuntu.com/cloud

MapReduce programming model. The TPlatform framework is composed by three layers (Figure 6):

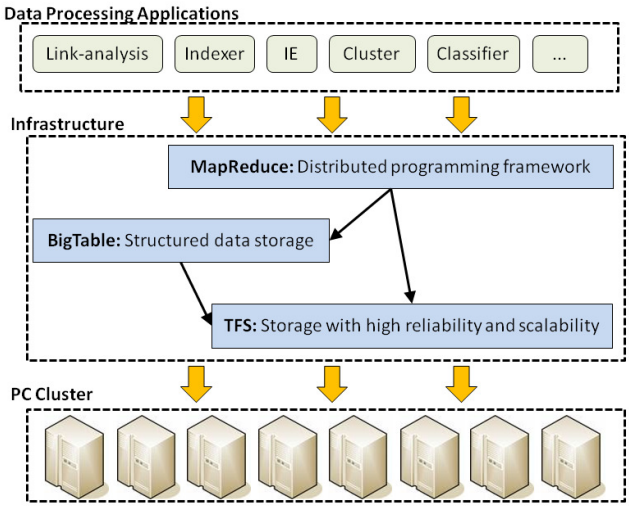

Figure 6. TPlatform framework [Peng et al 2009]

- **PC Cluster**: this layer provides the hardware infrastructure for data processing.
- **Infrastructure**: this layer consists of file system (TFS), distributed data storage mechanism (BigTable), and programming model (MapReduce).
- **Data Processing Applications**: this layer provides the services for users to develop their application (e.g. web data analysis and language processing).

4.6. Apache Virtual Computing Lab (VCL)

Apache VCL [VCL 2010] is an open-source solution for the remote access over the Internet to dynamically provision and reserve computational resources for diverse applications, acting as Software as a Service (SaaS) solution. VCL has a simple architecture formed by three tiers:

- **Web server**: represents the VCL portal and uses Linux/Apache/PHP solution. This portal provides an user interface that enable the requesting and management of VCL resources;
- **Database server**: storages information about VCL reservations, access controls, machine and environment inventory. It uses Linux/SQL solution;
- **Management nodes**: is the processing engine. A management node controls a subset of VCL resources, which may be physical blade servers, traditional rack, or virtual machines. It uses Linux/VCLD (perl)/image library solution. VCLD is a middleware responsible to process reservations or jobs assigned by the VCL web portal. According to type of environment requested, VCLD should assure that service (computational environment) will be available to user.

Figure 7 shows a conceptual overview of the VCL, where the user must connect firstly to the VCL Scheduling Application in order to access its resources through a web interface. Users may request a reservation to use the environment immediately or schedule to use it in the future.

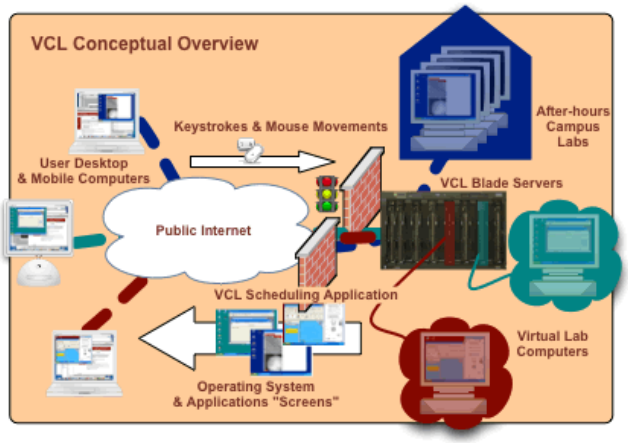

Figure 7. Apache conceptual overview [VCL 2010]

4.7. Enomaly Elastic Computing Platform

Enomaly ECP Community Edition under the AGPL license [Enomaly 2009] is the open source cloud solution offered by Enomaly Inc. This version focuses on virtual machine administration in small clouds environments. Compared with the Enomaly commercial solution (called Service Provider Edition), the Enomaly open source edition suffers from many restrictions, such as limited scalability, no capacity control mechanism, no support for accounting and metering, and so on.

5. Discussion

As pointed out earlier in this paper, there are several solutions for cloud computing[7] focusing on different areas and ranging from hardware resource outsourcing to user services providing. Each solution presents a different vision about cloud architecture and implementation. Moreover, each approach has an implication that directly impacts its business model: the closer to the hardware level, the more options a user can handle but at the cost of having to configure her cloud (more configuration flexibility).

Amazon EC2 and IBM Capacity on Demand (CoD) are solutions that offer to their users this configuration flexibility. In this business model, users can choose and configure computational resources at the hardware level and OS levels. At the other extreme, solutions like Google App Engine and Windows Azure, try to turn development easy to their users, but at the same time, confine them to specific APIs and software platforms. Moreover, solutions like JoliCloud[8] are more limited as they offer a single service (operating system). In the middle, there are solutions that offer a middleware-like approach to users, where the hardware resources can be configured and handled subject to some restrictions and where applications can also be developed.

All the presented open-source solutions and the cited commercial solutions are categorized into Figure 8. The graphic compares solutions and their business model (hardware, middleware and user level) according to configuration flexibility.

---

[7] More than 500 Cloud solutions have been reported.

[8] http://www.jolicloud.com/

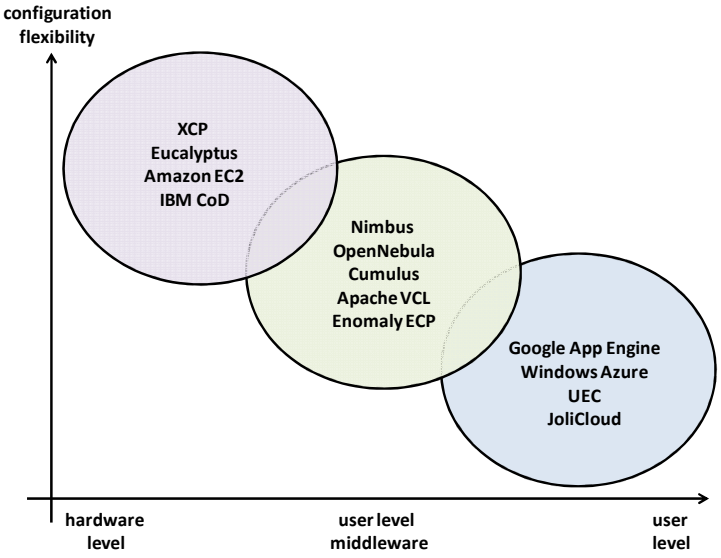

Figure 8. Cloud computing solutions

Finally, Table 1 presents a comparative board of the open source cloud solutions described in this paper, in terms of the service type (IaaS, PaaS, and SaaS), the main characteristics, and the infrastructure technologies. The table also cites some users of each cloud solution.

Table 1. Comparison between open-source Cloud Computing solutions

Solutions	Service	Main Characteristics	Infrastructure	Used by
XCP	IaaS	Only a tool for automatic maintenance of clouds	Xen	XCP community
Nimbus	IaaS	Aims to turn legacy clusters into IaaS Clouds	Xen hypervisor and KVM	Brookhaven National Labs[9]
OpenNebula	IaaS	Policy-driven resource allocation	Xen hypervisor	Cumulus Project
Eucalyptus	IaaS	Hierarchical Architecture	Xen hypervisor and KVM	UEC
TPlatform	PaaS	Focus on web text mining applications	TFS, BigTable and MapReduce	TPlatform Project[10]
Apache VCL	SaaS	Internet access for several applications	VMware	Educational[11] and Government[12] users
Enomaly	IaaS	Open version is focused in small clouds	Xen, KVM and VirtualBox	Several companies[13]

[9] http://www.bnl.gov/rhic/default.asp

[10] http://net.pku.edu.cn/~webg/tplatform/

[11] East Carolina University, Johnston Community College, North Carolina Central University, University of North Carolina at Greensboro, Wake Technical Community College and Western Carolina University

[12] North Carolina Community College System

[13] Some companies that use Enomaly Service Provider Edition: Orange/France Télécom, Bank of China, City Network, CentriLogic

6. Conclusions and Future Works

There is a clear need for the standardization of current cloud platforms at least of terms of interface, negotiation and access through Web services. Understandably, this is a considerable task as many clouds use different abstraction levels, some are generic whereas others focus on a specific application domain, etc. Some initial steps have been taken into this direction with the setup of the Open Cloud Manifesto, an initiative supported by hundreds of companies. In the mean time we will continue to see some clouds such as Nimbus implementing a number of front ends (e.g. Amazon EC2 and WSRF) to ensure access to their existing users.

Interestingly, some solutions such as the OpenNebula have been first to adopt policies for resource management. The use of policies remains a challenge in many areas and clouds may benefit from it. It is also important to acknowledge the leadership and string presence of academic efforts such as Eucalyptus and Xen. These have been at the forefront of inovation supported by the many commercial cloud systems that currently are based on these. These efforts are expected to continue as more work is needed to remove much of the misticism and conflicts surrounding the use of clouds.

As future work, authors proposes a quantitative comparison of the presented solutions through performance evaluation measurements.

References

Amazon. (2006) “Amazon simple storage service (Amazon S3) – API Reference”, http://docs.amazonwebservices.com/AmazonS3/2006-03-01/.

Andersen, D. (2002) “Theoretical Approaches to Node Assignment”, Unpublished manuscript. http://www.cs.cmu.edu/dga/papers/andersen-assign.ps

Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R.H., Konwinski, A., Lee, G., Patterson, D.A., Rabkin, A., Stoica, I., Zaharia, M. (2009) “Above the Clouds: A Berkeley View of Cloud Computing”, Tech. Rep. UCB/EECS-2009-28, EECS Department, University of California, Berkeley.

Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I. and Warfield, A. (2003) “Xen and the art of virtualization”. In: 19th ACM symposium on Operating systems principles, New York, NY, USA.

Buyya, R., Yeo, C.S., Venugopal, S., Broberg, J., Brandic, I. (2009) “Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility”. In: Future Generation Computer Systems, Elsevier B. V.

Chowdhury, N.M. M. K. and Boutaba, R. (2010) “A survey of network virtualization”. In: Computer Networks, volume 54, issue 5, pages 862-876. Elsevier B. V.

Citrix Systems. (2010) “Xen Cloud Platform - Advanced Virtualization Infrastructure for the Clouds”, http://www.xen.org/products/cloudxen.html.

Citrix Systems. (2010) “Xen Hypervisor”, http://xen.org/products/xenhyp.html.

Citrix Systems. (2010) “The Xen Cloud Project”, The Citrix Blog. Available at http://community.citrix.com/pages/viewpage.action?pageId=81690805

Enomaly. (2009) “Enomaly – Elastic Computing”. http://src.enomaly.com/.

Keahey, K. (2009) “Nimbus: Open Source Infrastructure-as-a-Service Cloud Computing Software”, Workshop on adapting applications and computing services to multi-core and virtualization, CERN, Switzerland.

McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S. and Turner, J. (2008) “OpenFlow: enabling innovation in campus networks”, ACM SIGCOMM Computer Communication Review.

Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G., Soman, S., Youseff, L. and Zagorodnov, D. (2009) “The Eucalyptus Open-Source Cloud Computing System”. In: 9th IEEE/ACM International Symposium on Cluster Computing and the Grid.

OpenCloud. (2009) “The Open Could Manifesto”. http://www.opencloudmanifesto.org/.

OpenCloud. (2010) “Cloud Computing Use Cases Whitepaper - Version 3.0”. http://groups.google.com/group/cloud-computing-use-cases?hl=en.

OpenNebula Project. (2010) “OpenNebula.org – The Open Source Toolkit for Cloud Computing”, http://www.opennebula.org/.

Peng, B., Cui, B. and Li, X. (2009) “Implementation Issues of A Cloud Computing Platform”. IEEE Data Engineering Bulletin, volume 32, issue 1.

Vaquero, L.M., Merino, L. R. Caceres, J. Lindner, M. (2008) “A break in the clouds: towards a cloud definition”. ACM SIGCOMM Computer Communication Review.

VCL. (2010). “Virtual Computing Lab”. https://cwiki.apache.org/VCL/.

Vouk, M.A. (2008) “Cloud Computing – Issues, Research and Implementations”. In: Journal of Computing and Information Technology. University of Zagreb, Croatia.

Xen.org. (2009) “Xen Cloud Platform Administrator’s Guide: Release 0.1”.

Wang, L., Tao, J., Kunze, M., Rattu, D. and Castellanos, A. (2008) “The Cumulus Project: Build a Scientific Cloud for a Data Center”. In: Workshop on Cloud Computing and Its Applications, Chicago, USA.