A Taxonomic Specification of Cloud@Home

Salvatore Distefano, Vincenzo D. Cunsolo, and Antonio Puliafito

University of Messina,
Contrada di Dio, S. Agata, 98166 Messina, Italy
{sdistefano,vdcunsolo,apuliafito}@unime.it

Abstract. In this paper we present the *Cloud@Home* paradigm as an effective solution to the problem of building open and interoperable Clouds. In this new paradigm, users' hosts are not passive interface to Cloud services anymore, but they can interact (for free or by charge) with other Clouds, that therefore must be able to interoperate. Such innovative paradigm merges goals of Cloud and Volunteer computing by aggregating the computational potentialities of many small, low power systems, exploiting the long tail effect of computing. This paper, starting from a well-known Cloud layered taxonomy, try to implement such goals into a specific logical architecture and the corresponding middleware core structure, therefore deployed into a physical infrastructure.

1 Introduction

Cloud computing is derived from the *service-centric perspective* that is quickly and widely spreading on the IT world. From this perspective, all capabilities and resources of a Cloud (usually geographically distributed) are provided to users *as a service*, to be accessed through the Internet without any specific knowledge of, expertise with, or control over the underlying technology infrastructure that supports them.

In order to achieve such goals it is necessary to implement a level of abstraction of physical resources, uniforming their interfaces and providing means for their management, adaptively to user requirements. This is done through *virtualizations* [1], *service mashups* (Web 2.0) [2] and *service oriented architectures* (SOA) [3]. An interesting attempt to fix Cloud concepts and ideas is provided in [4] thorough an ontology which demonstrates a dissection of the Cloud into the five main layers shown in Fig. 1, where higher layers services can be composed from the services of the underlying layers, which are:

1. *Cloud Application Layer:* provides interface and access management tools (Web 2.0, authentication, billing, SLA, etc.), specific application services, services mashup tools, etc. to the Cloud end users. This model is referred to as Software as a Service (SaaS).
2. *Cloud Software Environment Layer*: providers of the Cloud software environments supply the users Cloud applications' developers with a programming-language-level environment with a set of well-defined APIs. The services provided by this layer are referred to as Platform as a Service (PaaS).

D.-S. Huang et al. (Eds.): ICIC 2010, LNAI 6216, pp. 527–534, 2010.

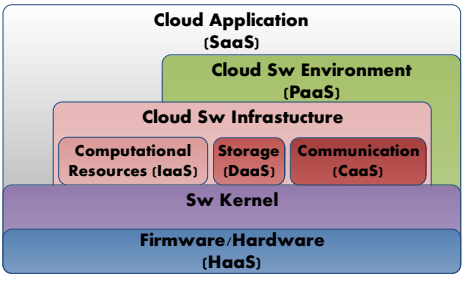

Fig. 1. Cloud Ontology-Taxonomy

3. *Cloud Software Infrastructure Layer*: provides fundamental resources to other higher-level layers. Services can be categorized into:
   - *Computational resources* - provides computational resources (VM) to Cloud end users. Often, such services are dubbed Infrastructure as a Service (IaaS).
   - *Data storage* - allows users to store their data at remote disks and access them anytime from any place. These services are commonly known as Data-Storage as a Service (DaaS)
   - *Communications* provides some communication capabilities that are service oriented, configurable, schedulable, predictable, and reliable. Towards this goal, the concept of Communication as a Service (CaaS) emerged to support such requirements, as well as network security, dynamic provisioning of virtual overlays for traffic isolation or dedicated bandwidth, guaranteed message delay, communication encryption, and network monitoring.

   SOAP and REST are examples of interface protocols used with some Cloud computational resources.
4. *Software Kernel*: provides the basic software management for the physical servers that compose the Cloud. OS kernel, hypervisor, virtual machine monitor, clustering and grid middleware, etc.
5. *Hardware and Firmware*: form the backbone of the Cloud. End users directly interacting with the Cloud at this layer are normally big enterprises with huge IT requirements in need of subleasing Hardware as a Service (HaaS).

A great interest on Cloud computing has been manifested from both academic and private research centers, and numerous projects from industry and academia have been proposed, such as: Amazon EC^2 [5], Microsoft Azure Services Platform [6], Reservoir [7], Eucalyptus [8], etc. All of them support and provide an on-demand computing paradigm, in the sense that a user submits his/her requests to the Cloud that remotely, in a distributed fashion, processes them and gives back the results. This client-server model well fits aims and scopes of commercial Clouds: the business. But, on the other hand, it represents a restriction for scientific Clouds, that have a view closer to *Volunteer computing*.

Volunteer computing (also called *Peer-to-Peer computing*, *Global computing* or *Public computing*) uses computers volunteered by their owners, as a source of computing power and storage to provide distributed scientific computing [9].

We believe the Cloud computing paradigm is applicable also at lower scales, from the single contributing user, that shares his/her desktop, to research groups, public administrations, social communities, small and medium enterprises, which make available their distributed computing resources to the Cloud. Both free sharing and pay-per-use models can be easily adopted in such scenarios.

We propose a more "democratic" form of Cloud computing, in which the computing resources of single users accessing the Cloud can be shared with the others, in order to contribute to the elaboration of complex problems. Since this paradigm is very similar to the Volunteer computing one it can be named *Cloud@Home*. Both hardware and software compatibility limitations and restrictions of Volunteer computing can be solved in Cloud computing environments, allowing to share both hardware and software resources. The Cloud@Home paradigm could be also applied to commercial Clouds, establishing an *open computing-utility market* where users can both buy and sell their services.

In order to achieve and implement all these goals through the Cloud@Home paradigm, in section 2 we firstly discuss and investigate aims and requirements, and then in section 3 we describe the architecture of the Cloud@Home infrastructure. Section 4 concludes the paper recapitulating our work and discussing about challenges and future work.

2 The Cloud@Home Idea

The idea behind Cloud@Home is to reuse *"domestic"* computing resources to build voluntary contributors' Clouds that can interoperate each other and, moreover, with other, also commercial, Cloud infrastructures. By Cloud@Home anyone can experience the power of Cloud computing, both actively providing his/her own resources and services, and/or passively submitting his/her applications.

In Cloud@Home both the commercial/business and the volunteer/scientific viewpoints coexist: in the former case the end user orientation of Cloud is extended to a collaborative two-way Cloud in which users can buy and/or sell their resources/services; in the latter case, the Grid philosophy of few but large computing requests is extended and enhanced to *open* Virtual Organizations. In both cases QoS requirements could be specified, introducing in the Grid and Volunteer philosophy (*best effort*) the concept of quality.

Cloud@Home can be also considered as a generalization and a maturation of the @home philosophy: a context in which users voluntarily share their resources without any compatibility problem. This allows to knock down both hardware (processor bits, endianness, architecture, network) and software (operating systems, libraries, compilers, applications, middlewares) barriers of Grid and Volunteer computing. Moreover, in Cloud@Home the term resources must be interpreted in the more general Cloud sense of services. This means that

Cloud@Home allows users to share not only physical resources, as in @home projects or in Grid environments, but any kind of service. The *flexibility* and the *extendibility* of Cloud@Home could allow to easily arrange, manage and make available (for free or by charge) significant computing resources (greater than in Clouds, Grids and/or @home environments) to everyone that owns a computer.

On the other hand, Cloud@Home can be considered as the enhancement of the Grid-Utility vision of Cloud computing. In this new paradigm, user's hosts are not passive interfaces to Cloud services, but they can be actively involved in the computing. Stressing this concept at worst, single nodes and services can be potentially enrolled by the Cloud@Home middleware, in order to build own-private Cloud infrastructures that can (for free or by charge) interact with other Clouds.

The overall infrastructure must deal with the high dynamics of its nodes/ resources, allowing to move and reallocate data, tasks and jobs. It is therefore necessary to implement a lightweight middleware, specifically designed to optimize migrations. The choice of developing such middleware on existing technologies (as done in Nimbus-Stratus starting from Globus) could be limitative, inefficient or not adequate from this point of view. This represents another significant enhancement of Could@Home against Grid: a lightweight middleware allows to involve limited resources' devices into the Cloud, mainly as consumer hosts accessing the Cloud through "thin client", but also, in some specific applications, as contributing hosts implementing (light) services according to their availabilities. Moreover, the Cloud@Home middleware does not influence the code writing as Grid and Volunteer computing paradigms do.

Another important goal of Cloud@Home is the *security*. Volunteer computing has some lacks in security concerns, while the Grid paradigm implements complex security mechanisms. The virtualization in Clouds implements the isolation of the services, but does not provide any protection from local access. With regards security, the specific goal of Cloud@Home is to extend the security mechanisms of Clouds to the protection of data from local access. Since Cloud@Home is composed of an amount of resources potentially larger than commercial or proprietary Cloud solutions, its *reliability* can be compared to the Grid or the Volunteer computing one, and it should be greater than other Clouds.

Last but not least, *interoperability* is one of the most important goal of Cloud@Home. Cloud interoperability will enable Cloud infrastructures to evolve into a worldwide, transparent platform in which applications are not restricted to enterprise Clouds and Cloud service providers. We must build new standards and interfaces that will enable enhanced portability and flexibility of virtualized applications. Up to now, significant discussion has occurred around open standards for cloud computing. In this context, the "Open Cloud Manifesto" (www.opencloudmanifesto.org) provides a minimal set of principles that will form a basis for initial agreements as the Cloud community develops standards for this new computing paradigm. In Grid environment interoperability is a very tough issue, many people tried to address it for many years and still we are far

away to solve the problem. Interoperability in Cloud contexts is easier, since virtualization avoids the major architectural, physical, hardware and software problems. Anyway, problems of compatibility among different virtual machines (VM) monitors can arise and therefore must be adequately faced, as the Open Virtualization Format (OVF) [1] group is trying to do.

The most important issues that should be taken into account in order to implement Cloud@Home can be synthesized as follows:

- *Resources and Services management* - a mechanism for managing resources and services offered by Clouds is mandatory.
- *Frontend* - abstraction is needed in order to provide users with a high level service oriented point of view of the computing system.
- *Security* - effective mechanisms are required to provide: authentication, resources and data protection, data confidentiality and integrity.
- *Interoperability among Clouds* - it should be possible for Clouds to interoperate each other.
- *Business models* - it is mandatory to provide QoS and SLA management for both commercial and open volunteer Clouds (traditionally best effort).

3 The Cloud@Home Architecture

3.1 Logical Abstract Model

A possible Cloud@Home architecture that could accomplish the issues above listed is shown in Fig. 2, adapted to the ontology provided in [4] and reported in Fig. 1. Two types of users are distinguished in such architecture according to the role they assume in the Cloud: *end users*, if they only interface the Cloud for submitting requests, and/or *contributing users* if they make available their resources and services for building up and supporting the Cloud. According to this point of view, the Cloud is composed of several *contributing hosts* offered by the corresponding contributing users to end users that interact with the Cloud

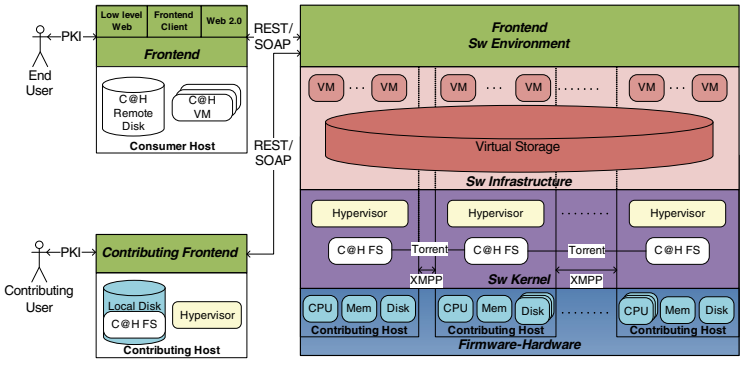

Fig. 2. Cloud@Home Logical-Abstract Model

and submit their requests through their *consumer hosts*. To access a Cloud, both contributing and end users must authenticate themselves into the system. One of the main enhancement of Cloud@Home is that a user/host can be at the same time both contributing and end user/consumer host, establishing a symbiotic mutual interaction with the Cloud.

Such architecture will be described in the following by identifying and detailing tasks and functions of each of the five layers characterized in the Cloud ontology of section 1.

Software Environment - The Cloud@Home software environment implements the user-infrastructure frontend interface. It is responsible for the resources and services management (enrolling, discovery, allocation, coordination, monitoring, scheduling, etc.) from the global Cloud system's perspective. It also provides tools, libraries and API for translating user requirements into physical resources' demand. Moreover, in commercial Clouds, it must be able to negotiate the QoS policy to be applied (SLA), therefore monitoring for its fulfillment and, in case of unsatisfactory results, adapting the computing workflow to such QoS requirements. If the Cloud's available resources can not satisfy the requirements, the frontend provides mechanisms for requesting further resources and services to other Clouds, both open and/or commercial. In other words, the Cloud@Home frontend implements the interoperability among Clouds, also checking for services' reliability and availability.

Software Infrastructure - Two basic services are provided by the software infrastructure to the software environment and, consequently, to end users: *execution* and *storage* services. The execution service allows to create and manage virtual machines. A user, sharing his/her resources within a Cloud@Home, allows the other users of the Cloud to execute and manage virtual machines locally at his/her node, according to policies and constraints negotiated and monitored through the software environment. The storage service implements a storage system distributed across the storage hardware resources composing the Cloud, highly independent of them since data and files are replicated according to QoS policies and requirements to be satisfied.

Software Kernel - The software kernel provides to the software infrastructure, mechanisms and tools for locally managing the Cloud physical resources in order to implement execution and storage services. Cloud@Home negotiates with users that want to join a Cloud about his/her contribution. This mechanism involves the software kernel that provides tools for reserving execution and/or storage resources for the Cloud, and monitors these resources, such that constraints, requirements and policies thus specified are not violated.

Firmware/Hardware - The Cloud@Home firmware/ hardware layer is composed of a *"cloud"* of generic contributing nodes and/or devices geographically distributed across the Internet. They provide to the upper layers the physical-hardware resources for implementing the execution and storage services.

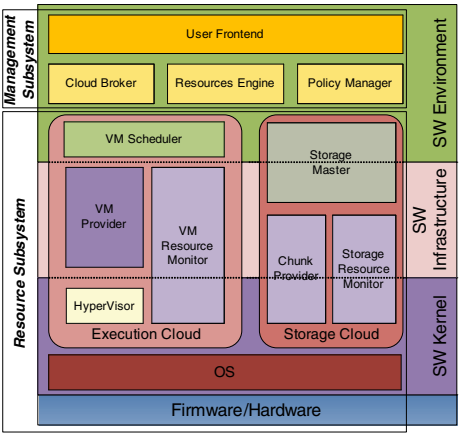

Fig. 3. Cloud@home Core Structure Organization

3.2 Core Architecture

The blocks implementing the functions above identified are pictorially depicted in the layered model of Fig. 3, that reports the core structure of the Cloud@Home server-side middleware. It is subdivided into two subsystems: *management* subsystem implementing the upper layer of the functional architecture, and *resource subsystem* implementing lower level functionalities.

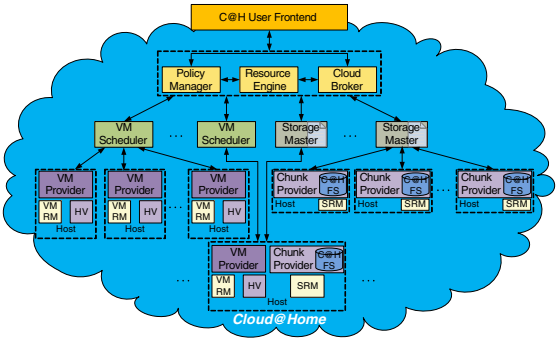

Fig. 4. Cloud@home Core Structure Infrastructure Deployment

Fig. 4 pictorially depicts the deployment of the Cloud@Home core structure into the physical infrastructure. On top of such hierarchy there are the blocks implementing the management subsystem, that can be deployed into different servers/nodes, one for each block, or can be grouped into the same node. Below them, VM schedulers and storage masters manage smaller groups (grid, clusters, multi-core nodes, etc) of resources. At the bottom of the hierarchy there are the contributing hosts. Each of such leaves contains the blocks, the software for supporting the specific service for what was enrolled into the Cloud.

4 Conclusions

In this paper we discussed an innovative computing paradigm merging volunteer contributing and Cloud approaches into Cloud@Home. This proposal represents a solution for building Clouds, starting from heterogeneous and independent nodes, not specifically conceived for this purpose. Cloud@Home implements a generalization of both Volunteer and Cloud computing by aggregating the computational potentialities of many small, low power systems, exploiting the long tail effect of computing.

In this way Cloud@Home opens the Cloud computing world to scientific and academic research centers, as well as to public administration and communities, till, potentially, single users: anyone can voluntarily support projects by sharing his/her resources. On the other hand, it opens the utility computing market to the single user that wants to sell his/her computing resources.

The paper try to implement this broader vision by a specific logical architecture and the corresponding middleware core structure deployed into a physical infrastructure, based on a Cloud layered ontology. The implementation of the middleware, starting from this specifications is actually work in progress.

References

1. VMWare Inc., X.I.: The open virtual machine format whitepaper for ovf specification (2007), http://www.vmware.com/appliances/learn/ovf.html
2. O'Reilly, T.: What is WEB 2.0 tml (September 2005), http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html
3. MacKenzie, C.M., Laskey, K., McCabe, F., Brown, P.F., Metz, R., Hamilton, B.A.: Reference Model for Service Oriented Architecture 1.0. OASIS SOA Reference Model Technical Committee (2006), http://docs.oasis-open.org/soa-rm/v1.0/
4. Youseff, L., Butrico, M., Da Silva, D.: Toward a unified ontology of cloud computing. In: Grid Computing Environments Workshop, GCE '08, November 2008, pp. 1–10 (2008)
5. Amazon Inc.: Elastic Compute Cloud. Amazon (November 2008), http://aws.amazon.com/ec2
6. Co., M.: Azure services platform, http://www.microsoft.com/azure/default.mspx
7. Reservoir Consortium: Reservoir Project (2009), http://www-03.ibm.com/press/us/en/pressrelease/23448.wss/
8. Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G., Soman, S., Youseff, L., Zagorodnov, D.: The eucalyptus open-source cloud-computing system. In: Proceedings of Cloud Computing and Its Applications (October 2008)
9. Anderson, D.P., Fedak, G.: The computational and storage potential of volunteer computing. In: CCGRID '06, pp.73–80 (2006)
10. Foster, I.: There's Grid in them thar Clouds. Ian Foster's blog (January 2008), http://ianfoster.typepad.com/blog/2008/01/theres-grid-in.html