2009 Eighth IEEE International Symposium on Network Computing and Applications

Volunteer Computing and Desktop Cloud: the Cloud@Home Paradigm

Vincenzo D. Cunsolo, Salvatore Distefano, Antonio Puliafito and Marco Scarpa University of Messina,
Contrada di Dio, S. Agata, 98166 Messina, Italy.
Email: vdcunsolo,sdistefano,apuliafito,mscarpa@unime.it

Abstract—Only commercial Cloud solutions have been implemented so far, offering computing resources and services for renting. Some interesting projects, such as Nimbus, OpenNEbula, Reservoir, work on Cloud. One of their aims is to provide a Cloud infrastructure able to provide and share resources and services for scientific purposes. The encouraging results of Volunteer computing projects in this context and the flexibility of the Cloud, suggested to address our research efforts towards a combined new computing paradigm we named Cloud@Home. On one hand it can be considered as a generalization of the @home philosophy, knocking down the barriers of Volunteer computing, and also allowing to share more general services. On the other hand, Cloud@Home can be considered as the enhancement of the grid-utility vision of Cloud computing. In this new paradigm, users' hosts are not passive interface to Cloud services anymore, but they can interact (free or by charge) with other Clouds.

In this paper we present the Cloud@Home paradigm, highlighting its contribution to the actual state of the art on the topic of distributed and Cloud computing. We detail the functional architecture and the core structure implementing such paradigm, demonstrating how it is really possible to build up a Cloud@Home infrastructure.

1 INTRODUCTION

Cloud computing is a distributed computing paradigm that mixes aspects of *Grid computing*, *Internet computing*, *Utility computing*, *Autonomic computing* and *Green computing*. The development and the success of Cloud is due to the maturity reached by both hardware and software *virtualization*. This made realistic the L.Kleinrock outlook [12] of computing as the 5th utility.

Cloud computing comes from the *service-centric perspective* that is widely spreading on the IT world. From this perspective, all capabilities and resources of a Cloud 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 supporting them. It offers a user-centric interface that acts as a unique, user friendly, point of access for users' needs and requirements. Moreover, Cloud computing provides *on-demand service provision*, *QoS guaranteed offer*, and *autonomous system* for managing hardware, software and data transparently to users [18].

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*, *service mashups* (Web 2.0) and *service oriented architectures* (SOA).

Virtualization allows to execute a software version of a hardware machine into a host system, in an isolated way. It "homogenizes" resources: problems of compatibility are overcome by providing heterogeneous hosts of a distributed computing environment (the Cloud) with the same virtual machine. The software implementing virtualization is named *hypervisor*.

The Web 2.0 provides a uniform interface to Cloud services, implementing service mashup. It is mainly based on an evolution of JavaScript with improved language constructs (late binding, clousers, lambda functions, etc) and AJAX interactions.

SOA is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. In SOA, *services* are the mechanism by which needs and capabilities are brought together. SOA defines standard interfaces and protocols that allow developers to encapsulate information tools as services that clients can access without knowledge of, or control over, their internal workings [8].

A great interest on Cloud computing has been manifested and numerous projects from industry and academia have been proposed. In commercial contexts, among the others we highlight: Amazon Elastic Compute Cloud [1] IBM's Blue Cloud [11], Sun Microsystems Network.com [15], Microsoft Azure Services Platform [5]. Scientific open activities and projects are: Reservoir [14] Nimbus/Stratus/Wispy/Kupa [17] and OpenNEbula [7]. All of them support and provide an on-demand computing paradigm: users submit their 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 uses computers volunteered by their owners, as a source of computing power and storage to provide distributed scientific computing [3]. It is behind the *"@home"* philosophy of sharing/donating network connected resources for supporting distributed scientific computing.

978-0-7695-3698-9/09 $25.00 © 2009 IEEE
DOI 10.1109/NCA.2009.41

We believe that the Cloud computing paradigm is applicable also at lower scales, from the single contributing user, that shares his/her desktop/devices according to the available capabilities, 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.

From the utility point of view, the rise of the "techno-utility complex" and the corresponding increase of computing resources demand, in some cases growing dramatically faster than Moore's Law as predicted by the Sun CTO Greg Papadopoulos in the *red shift theory* for IT [13], could bring, in a close future, towards an *oligarchy*, a lobby or a trust of few big companies controlling the whole computing resources market.

To avoid such scenario, we suggest to address the problem in a different way: instead of building costly private *data centers*, that the Google CEO Eric Schmidt likes to compare to the prohibitively expensive cyclotrons [4], we propose a more "democratic" form of Cloud computing, in which the computing resources of single users 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*. Compatibility limitations of Volunteer computing can be solved in Cloud computing environments, allowing to share resources and services.

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. Since the computing power can be described by a "long-tailed" distribution, in which a high-amplitude population (Cloud providers and commercial data centers) is followed by a low-amplitude population (small data centers and private users) which gradually "tails off" asymptotically, Cloud@Home can catch the *Long Tail* effect [2], providing similar or higher computing capabilities than commercial providers' data centers, by grouping small computing resources from many single contributors.

In the following we demonstrate how it is possible to make real all these aims through the Cloud@Home paradigm. Thus, in section 3 we describe the functional architecture of the Cloud@Home infrastructure, and in section 4 we characterize the blocks implementing the functions previously identified into the Cloud@Home core structure. Section 5 resumes the paper and discusses about challenges and future work.

2 WHY CLOUD@HOME?

The necessity of such a new computing paradigm is strictly related to the limits of existing Cloud solutions. For years Grid computing has been considered as the solution for all the computing problems: a secure, reliable, performing platform for managing geographically distributed resources. But it has some drawbacks: it is sensitive to hardware or software differences or incompatibility; it is not possible to dynamically extend a Virtual Organization (VO) by on-line enrolling resources, and consequently is not possible to share local resources, if they are not initially enrolled in the VO; it often does not face QoS and billing problems; it mainly supports *data parallelism* against *task parallelism*, making difficult the composition of services; users need to have knowledge of both the distributed system and the application requirements in order to submit and manage jobs.

These lacks have been partially faced and solved in Utility and Cloud computing, implementing service oriented paradigms with higher level user friendly interfaces, starting from on-demand models: users commission their computing, pay and get the results. Since they are mainly thought for commercial applications, QoS and business policies have to be carefully addressed. Utility and Cloud computing lack of an open, free viewpoint: as in the Grid computing, it is not possible to enroll resources or services, as also to build custom data centers by dynamically aggregating resources and services not conceived with this purpose. Moreover, each Cloud has its own interface and services, therefore it cannot communicate or interoperate with other Clouds.

On the other hand the Volunteer computing paradigm implements an open distributed environment in which resources (not services as in the Cloud) can be shared. But it manifests the same problem of Grid with regard to the compatibility among resources. Moreover, due to its purpose, it also does not implement any QoS and billing policy.

2.1 Aims and goals

Ian Foster summarizes the computing paradigm of the future as follows [9]: *"... we will need to support on-demand provisioning and configuration of integrated "virtual systems" providing the precise capabilities needed by an end-user. We will need to define protocols that allow users and service providers to discover and hand off demands to other providers, to monitor and manage their reservations, and arrange payment. We will need tools for managing both the underlying resources and the resulting distributed computations. We will need the centralized scale of today's Cloud utilities, and the distribution and interoperability of today's Grid facilities."*.

We share all these requirements, but in a slightly different perspective: we want to actively involve users into such a new form of computing, allowing to create own interoperable Clouds. We believe that it is possible to export, apply and adapt the *"@home"* philosophy to the Cloud computing paradigm. By merging Volunteer and Cloud computing, a new paradigm can be created: *Cloud@Home*. This new computing paradigm gives back the power and the control to users, who can decide how to manage their resources/services in a global, geographically distributed context. They can voluntarily sustain scientific projects by free placing their resources/services at the scientific research centers' disposal (eventually according to a credit mechanism), or they can earn money by selling their resources to Cloud computing providers in a pay-per-use/share context, according to their capabilities.

Therefore, in Cloud@Home both the commercial and the volunteer 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.

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

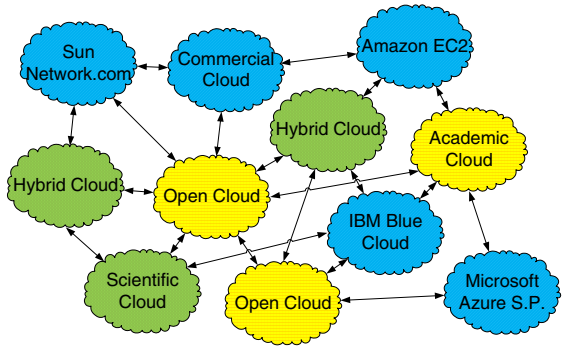

Figure 1. Cloud@Home Scenario

The Cloud@Home motto is: *heterogeneous hardware for homogeneous Clouds*. Thus, the scenario we prefigure is composed of several coexisting and interoperable Clouds, as pictorially depicted in Fig. 1. *Open Clouds* (yellow) identify Clouds operating for free; *Commercial Clouds* (blue) characterize entities or companies selling their computing resources for business; *Hybrid Clouds* (green) can both sell or give for free their services.

The overall infrastructure must deal with the high dynamism 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. It allows to involve limited resources' devices into the Cloud, and 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 data security, the specific goal of Cloud@Home is to extend the security mechanisms of Clouds to the protection of data from local access.

Last but not least, *interoperability* is one of the most important goal of Cloud@Home. This is an open problem in Grid, Volunteer and Cloud computing that we want to adequately face in Cloud@Home.

2.2 Application Scenarios

Different application scenarios can be imagined for Cloud@Home:

- Scientific research centers, communities - the Volunteer computing inspiration of Cloud@Home provides means for the creation of open, interoperable Clouds for supporting scientific purposes, overcoming the portability and compatibility problems highlighted by @home projects. Similar benefits could be experienced in public administrations and open communities (social network, peer-to-peer, etc).
- Enterprises - planting a Cloud@Home computing infrastructure in commercial locations can bring considerable benefits, especially in small and medium but also in big enterprises. It could be possible to implement own data center with local, existing, off the shelf, resources. The interoperability among Clouds allows to buy computing resources from commercial Cloud providers if needed or, otherwise, to sell the local Cloud computing resources to the same providers that can resell them. This allows to reduce and optimize business costs according to QoS/SLA policies, improving performances and reliability. For example, this paradigm allows to deal with the *flow peaks economy*: data centers could be sized for the medium case, and worst cases (peaks) could be managed by buying computing resources from Cloud providers.

3 Cloud@Home Overview

With Cloud@Home, anyone can experience the power of Cloud computing, both actively providing his/her own resources and services, and passively submitting his/her applications.

3.1 Issues, Challenges and Open Problems

In order to implement a lightweight Cloud@Home middleware, the following issues have to be taken into consideration:

- 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, and a unique, uniform access point to the Cloud.
- Security - effective mechanisms are required to provide: authentication, resources and data protection, data confidentiality and integrity.

- Reliability - it is necessary to implement redundancy of resources and services, and hosts' recovery policies.
- Interoperability - Clouds have to be able to interoperate each other.
- Business models - it is necessary to provide QoS and SLA management for both commercial and open volunteer Clouds in order to discriminate among the applications to be run.

3.2 Basic Architecture

A possible Cloud@Home architecture is shown in Fig. 2, identifying three hierarchical layers: *frontend*, *virtual* and *physical*.

3.3 Frontend Layer

The Cloud@Home frontend layer is responsible for the resources and services management (enrolling, discovery, allocation, coordination, monitoring, scheduling, etc) from the global Cloud system's perspective. The frontend layer provides tools for translating end-user requirements into physical resources' demand, also considering QoS/SLA constraints, if specified by the user. 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 available Cloud's resources and services can not satisfy the requirements, the frontend layer provides mechanisms for requesting further resources and services to other Clouds, both open and/or commercial. In other words, the Cloud@Home frontend layer implements the interoperability among Clouds. In order to improve reliability and availability of services and resources, especially if QoS policies and constraints have been specified, it is necessary introduce redundancy.

The frontend layer is split into two parts, as shown in Fig. 2: the server side, implementing the resources management and related problems, and the *light* client side, only providing mechanisms and tools for authenticating, accessing and interacting with the Cloud.

The frontend layer provides means, tools and policies for managing users. The best mechanism to achieve secure authentications is the *Public Keys Infrastructure* (PKI) [16], better if combined with smartcard devices that, through a trusted certification authority, ensure the user identification and provide Single Sign-on.

3.4 Virtual Layer

The virtualization of physical resources offers to end-users a homogeneous view of Cloud's services and resources. Two basic services are provided by virtual to frontend layer and, consequently, to end-users: *execution* and *storage* services.

The execution service is the tool for creating and managing 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 at the frontend layer. As shown in Fig. 2, from the end-user point of view an execution Cloud is seen as a set of virtual machines available and ready-to-use. The virtual machines' *isolation* implements protection and therefore security.

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. From the end-user point of view, a storage Cloud appears as a locally mounted remote disk.

In a distributed environment where any users can host part of private data, it is necessary to protect such data from unauthorized accesses. A way to obtain data *confidentiality* and *integrity* could be the cryptography, as better explained in the physical layer description.

3.5 Physical Layer

The physical layer is composed of a *"cloud"* of generic nodes and/or devices geographically distributed across the Internet. They provide to the upper virtual layer both physical resources for implementing execution and storage services and mechanisms and tools for locally managing such resources.

Cloud@Home negotiates with users that want to join a Cloud about his/her contribution. This mechanism involves the physical layer that provides tools for reserving physical execution and/or storage resources for the Cloud, and monitors these resources, such that constraints, requirements and policies thus specified are not violated. This ensures reliability and availability of physical resources, avoiding to overload the local system and therefore reducing the risk of crashes.

To implement the execution service in a generic device or to enroll it into an execution Cloud, the device must have a hypervisor ready to allocate and run virtual machines, as shown in Fig. 2. If a storage service is installed into the device, a portion of the local storage system must be dedicated for hosting the Cloud data. In such cases, the Cloud@Home file system is installed into the devices' shared storage space.

At physical layer it is necessary to implement data security (integrity and confidentiality) also ensuring that stored data cannot be accessed by who physically hosts them. We propose an approach that combines the inviolability of the Public Key Infrastructure asymmetric cryptography and the speed of the symmetric cryptography. Data are firstly encrypted by the symmetric key, and then stored into the selected host with the symmetric key encrypted by the user private key. This ensures that only authorized users can decrypt the symmetric key and consequently can access data.

SSH, TLS, IPSEC and other similar transmission protocols could be used to manage the connection among nodes. However, since the data stored in a Cloud@Home storage are encrypted, it is not necessary to use a secure channel for data transfers, more performant protocol, such as BitTorrent [6], can be used. The secure channel is required for sending and receiving non-encrypted messages and data to/from remote hosts.

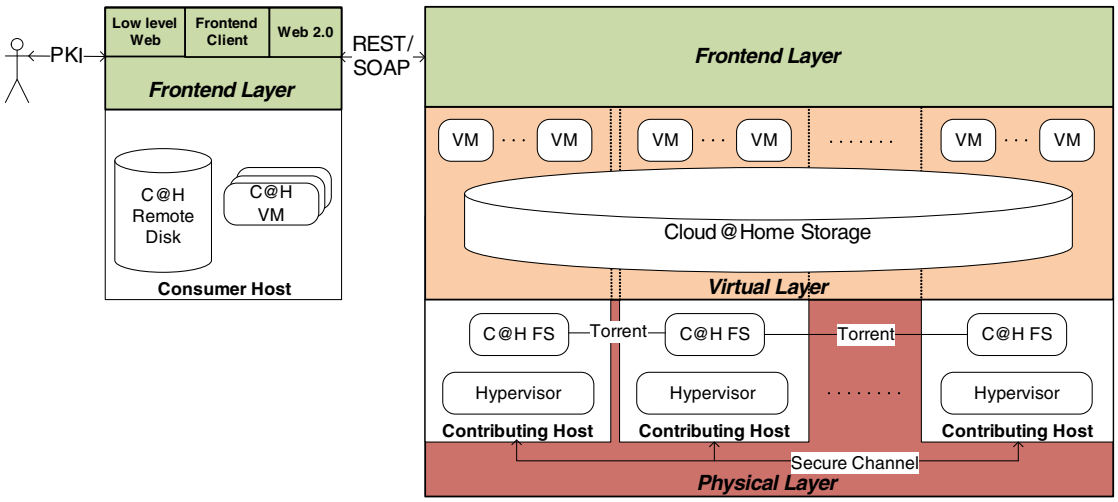

Figure 2. Basic architecture of Cloud@Home

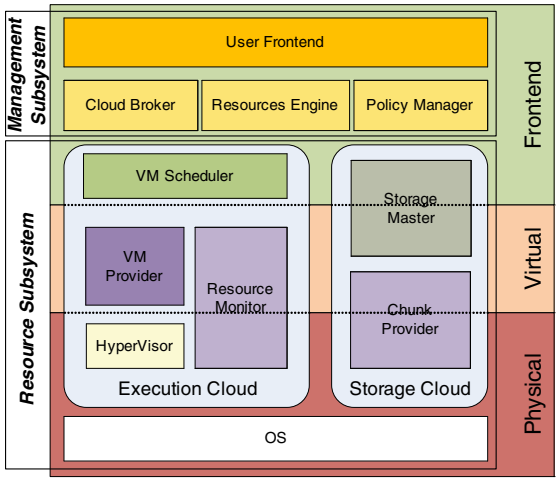

Figure 3. Cloud@home Core Structure Organization

4 Inside Cloud@Home

The blocks implementing the functional architecture of Cloud@Home above introduced, have been characterized in Fig. 3, reporting the core structure of the server-side system, subdivided into *management* and *resource subsystems*.

4.1 Management Subsystem

In order to enroll and manage the distributed resources and services of a Cloud, providing a unique point of access, it is necessary to adopt a centralized approach. This is implemented by the management subsystem that is composed of four parts: the *user frontend*, the *Cloud broker*, the *resource engine* and the *policy manager*.

The user frontend provides tools for Cloud@Home-User interactions. It collects and manages the users' requests issued by the client side. All such requests are transferred to the blocks composing the underlying layer for processing.

An important task carried out by the user frontend is the *Clouds interoperability*, implemented by point-to-point connecting the user frontend of the Clouds wishing to interoperate. In case one of the two Clouds to be point-to-point connected has not the Cloud@Home structure, it is necessary to translate the requests between Cloud@Home and foreign Clouds formats, task delegated by the user frontend to the Cloud broker. The Cloud broker collects and manages information about the available Clouds and the services they provide (both *functional* and *non-functional* parameters, such as QoS, costs, reliability, *request formats' specifications* for Cloud@Home-foreign Clouds translations, etc).

The policy manager provides and implements the Cloud's access facilities. This task falls into the security scope of identification, authentication and permission management, implemented starting from PKI as above introduced. The policy manager also manages the information about users' QoS policies and requirements.

The resource engine is the hearth of Cloud@Home. It is responsible for the resources' management. To meet this goal, the resource engine applies a hierarchical policy. It operates at higher level, in a centralized way, indexing all the resources of the Cloud. Incoming requests are delegated to *VM schedulers* or *storage masters* that, in a distributed fashion, manage the computing or storage resources respectively, coordinated by the resource engine.

4.2 Resource Subsystem

The resource subsystem contains all the blocks implementing the local and distributed management functionalities of Cloud@Home. This subsystem can be logically split into two parts offering different services over the same resources: the *execution Cloud* and the *storage Cloud*. The management subsystem merges them providing a unique Cloud that can offer both execution and/or storage services. The execution Cloud provides tools for managing virtual machines according

to users' requests and requirements coming from the management subsystem. It is composed of four blocks: *VM scheduler*, *VM provider*, *resource monitor* and *hypervisor*.

The VM Scheduler is a peripheral resource broker of Cloud@Home infrastructure, to which the resource engine delegates the management of Cloud resources and services. From the end-user point of view a VM is allocated somewhere on the Cloud, therefore its migration is transparent to end-users.

VM provider, resource monitor and hypervisor are responsible for managing a VM locally to a physical resource. A VM provider exports functions for allocating, managing, migrating and destroying a virtual machine on the corresponding host. The resource monitor allows to take under control the local computing resources according to requirements and constraints negotiated in the setup phase with the contributing-user. If during a virtual machine execution the local resources crash or become insufficient to keep running the virtual machine, the resource monitor asks the scheduler to migrate the VM elsewhere.

In order to implement a distributed storage system, the storage Cloud, we specify the *Cloud@Home file system* (FS), inspired by the Google FS [10]. The Cloud@Home FS splits data and files into *chunks* of fixed or variable size, depending on the storage resource available. The architecture of such file system is hierarchical: data chunks are physically stored on *chunk providers* and corresponding *storage masters* index the chunks.

The storage master is the directory server, indexing the data stored in the associated chunk providers. It directly interfaces with the resource engine to discover the resources storing data. The resource engine can be considered as the directory server indexing all the storage masters. To improve the storage Cloud reliability, storage masters must be replicated. Moreover, a chunk provider can be associated to more than one storage master.

Chunk providers physically store the data, that, as introduced above, are encrypted in order to achieve the confidentiality goal. Data reliability can be improved by replicating data chunks and chunk providers, consequently updating the corresponding storage masters. In this way, a corrupted data chunk can be automatically recovered and restored through the storage masters, without involving the end-user.

In order to avoid VM schedulers or storage masters become bottlenecks, once the VM/chunk providers have been located, VM interactions or data transfers are implemented by directly connecting end-users and VM or chunk providers, respectively.

5 Ready for Cloud@Home?

In this paper we proposed an innovative computing paradigm that represents a solution for building Clouds starting from heterogeneous and independent nodes, not specifically conceived for this purpose. This 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 communities or single users: anyone can voluntarily support projects by sharing his/her resources. On the other hand, it could open the utility computing market to the single user that wants to sell his/her computing resources. To realize this broader vision, several issues must be adequately taken into account: reliability, security, portability of resources and services, interoperability among Clouds, QoS/SLA and business models and policies.

It is necessary a common understanding, an ontology that fixes metrics and concepts such as resources, services and also overall Clouds functional and non-functional parameters, QoS, SLA, exposition format, and so on, that must be translated into specific interoperability standards. Fundamental aspect to take into account is the reliability: in a heterogeneous Cloud we can have highly reliable resources (NAS, computing servers), and barely reliable resources (temporary contributors). Cloud@Home must consider such parameter, specifying adequate management policies.

References

[1] Amazon Inc. Elastic Compute Cloud [URL]. Amazon, Nov. 2008. http://aws.amazon.com/ec2.

[2] Chris Anderson. *The Long Tail: How Endless Choice Is Creating Unlimited Demand*. Random House Business Books, July 2006.

[3] David P. Anderson and Gilles Fedak. The computational and storage potential of volunteer computing. In *CCGRID '06*, pages 73–80. IEEE Computer Society, 2006.

[4] Stephen Baker. Google and the Wisdom of Clouds. *BusinessWeek*, (December 24 2008), Dec. 2008. http://www.businessweek.com/magazine/content/07_52/b4064048925836.htm.

[5] Microsoft Co. Azure services platform. http://www.microsoft.com/azure/default.mspx.

[6] Bram Cohen. The BitTorrent Protocol Specification, 2008. http://www.bittorrent.org/beps/bep_0003.html.

[7] Distributed Systems Architecture Research Group. OpenNEbula Project [URL]. Universidad Complutense de Madrid, 2009. http://www.opennebula.org/.

[8] Ian Foster. Service-oriented science. *Science*, 308(5723):814–817, May 2005.

[9] Ian Foster. There's Grid in them thar Clouds. Ian Foster's blog, Jan. 2008. http://ianfoster.typepad.com/blog/2008/01/theres-grid-in.html.

[10] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. The Google File System. *SIGOPS*, 37(5):29–43, December 2003.

[11] IBM Inc. Blue Cloud project [URL]. IBM, June 2008. http://www-03.ibm.com/press/us/en/pressrelease/22613.wss/.

[12] Leonard Kleinrock. A vision for the internet. *ST Journal of Research*, 2(1):4–5, November 2005.

[13] Richard Martin. The Red Shift Theory. *InformationWeek*, (August 20 2007), Aug. 2007. http://www.informationweek.com/news/hardware/showArticle.jhtml?articleID=20180087

[14] Reservoir Consortium. Reservoir Project [URL], 2009. http://www-03.ibm.com/press/us/en/pressrelease/23448.wss/.

[15] Sun Microsystem. Network.com [URL]. SUN. http://www.network.com.

[16] S. Tuecke, V. Welch, D. Engert, L. Pearlman, and M. Thompson. Internet X.509 Public Key Infrastructure. RFC 3820, June 2004.

[17] University of Chicago-University of Florida-Purdue University-Masaryk University. Nimbus/Stratus/Wispy/Kupa Projects [URL], Jan. 2009. http://workspace.globus.org/clouds/nimbus.html/, http://www.acis.ufl.edu/vws/, http://www.rcac.purdue.edu/teragrid/resources/#wispy, http://meta.cesnet.cz/cms/opencms/en/docs/clouds.

[18] Lizhe Wang, Jie Tao, Marcel Kunze, Alvaro Canales Castellanos, David Kramer, and Wolfgang Karl. Scientific Cloud Computing: Early Definition and Experience. In *HPCC '08*, pages 825–830. IEEE Computer Society, 2008.