Journal of Applied Sciences, 2015
ISSN 1812-5654 / DOI: 10.3923/jas.2015.

# A Taxonomy and Survey of Distributed Computing Systems

Mohammadfazel Anjomshoa, Mazleena Salleh and Maryam Pouryazdanpanah Kermani
Faculty of Computing, Universiti Teknologi Malaysia, Malaysia

**Abstract:** Technology is the combination of knowledge and working hard. When users want to accomplish something using special technology, they do not want to know how it works. Technologies are coming to solve and ease our complex problems. It means that users only want to employ technology without any expert skills. Distributed computing is one of the technologies that is used to solve large and complex computational problems. It is based on distributed systems to address computational problems. In this study we are going to highlight the most well-known distributed computing paradigms and explain their technology building blocks. We provide a comprehensive explanation of cloud computing, volunteer computing and also volunteer cloud computing paradigm along with their advantages and also their open issues.

**Key words:** Cloud computing, volunteer computing, cloud@home, distributed computing, grid computing

## INTRODUCTION

Grid computing is a promising paradigm to enable High Performance Computing (HPC) (Buyya, 1999) by employing powerful computers as well as high speed network (Berman *et al*., 2003; Foster and Kesselman, 2004). In the 1990s, the term grid was established for users to employ computing power on demand (Foster *et al*., 2008). Since then, researchers developed grid idea in many different approaches. One of these approaches is Desktop Grid (DG) that uses idle desktop computers' resources (Fedak *et al*., 2001). Volunteer computing is a type of DGs that employs idle CPU cycles of host machine that donates publicly their computer's resources to solve a complex scientific problems (Anderson *et al*., 2005). Berkeley Open Infrastructure for Network Computing (BOINC) (Anderson, 2004) has emerged as the most well-known volunteer computing platform which provides 8.5 petaflops of processing power approximately and gathers more than 2,500,000 users all around the world. With the advent of cloud computing (Mell and Grance, 2011) which is the combination of many different technologies; virtualization, service-oriented arcitecture (Huhns and Singh, 2005) scalability, quality of service, utility computing, green computing, failover systems and cluster computing, the dream of computing has become reality (Rimal *et al*., 2009; Aversa *et al*., 2011). With the combination of cloud computing and volunteer computing a new computing paradigm has been coined which is dubbed Clouds@home (Distefano *et al*., 2010). The key concept of cloud computing is virtualization so to establish cloud-like infrastructure by using volunteer computing resources it is indispensable to adopt virtualization technology into volunteer computing frameworks.

This study intends to provide the comprehensive explanation of these computing paradigms.

## TAXONOMY

This section explains different forms of computing that are in common used today. Cloud computing, Volunteer computing and Clouds@home are addressed.

**Cloud computing:** Cloud computing is a computing paradigm that has considered as a revolution in computing. Finding a unique definition for cloud computing is not possible. Mell and Grance (2011) defines cloud computing as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction". Another definition for cloud computing is "Cloud computing refers to both the applications delivered as services over the internet and the hardware and systems software in the data centers that provide those services" (Armbrust *et al*., 2009, 2010).

**Corresponding Author:** Mohammadfazel Anjomshoa, Faculty of Computing, Universiti Teknologi Malaysia, Skudai, Johor, 81300, Malaysia

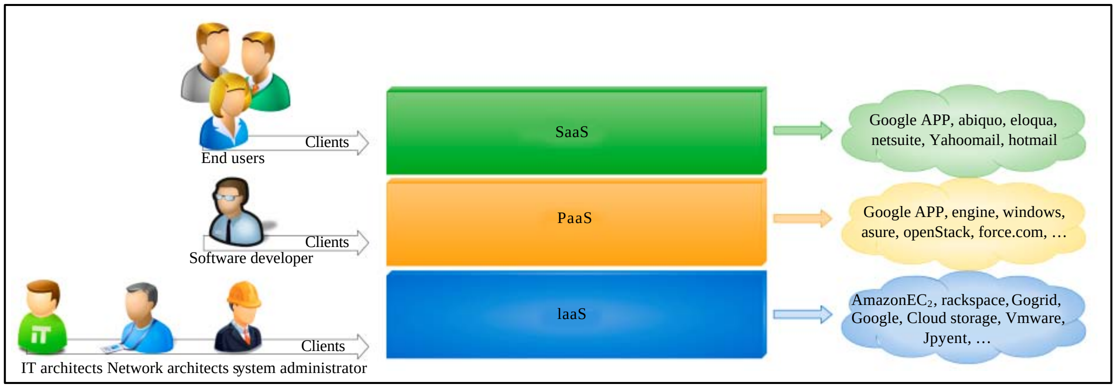

End users
Clients
Software developer
Clients
IT architects Network architects system administrator
Clients
SaaS
PaaS
laaS
Google APP, abiquo, eloqua, netsuite, Yahoomail, hotmail
Google APP, engine, windows, asure, openStack, force.com, …
AmazonEC$_2$, rackspace, Gogrid, Google, Cloud storage, Vmware, Jpyent, …

Fig. 1: Cloud computing architecture

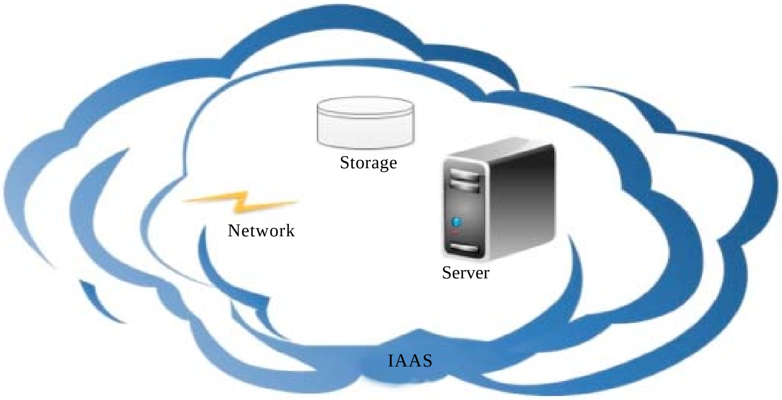


Fig. 2: IAAS components

According to the previous studies (Mell and Grance, 2011; Rimal *et al*., 2009) cloud computing has lots of features that can be addressed the recent problem of computing. Vaquero *et al*. (2009) defines cloud as form of grid computing, that virtual resources are dynamically allocated on a pay-per-used model.

Cloud computing has two important portions. Cloud computing deployment models and cloud computing services. Cloud computing has three type of models:

- **Private cloud:** Private clouds refer to the those clouds that data and process are managed from inside of organization
- **Public cloud:** Public clouds refer to those type of clouds that cloud infrastructure is available to the public and can be accessed via web
- **Hybrid cloud:** Hybrid clouds are the combination of multiple clouds (private, public, ...) with the goal of portability between different clouds that requires standardization technology

Cloud computing is based on service-oriented-architecture that makes all resources on the cloud as a service (Tsai *et al*., 2010). As Fig. 1 illustrates, general form of services in cloud computing are, infrastructure as a service-IaaS, platform as a service-PaaS, software as a service-SaaS (Mell and Grance, 2011). These levels are supported by virtualization and management tools.

Cloud computing infrastructure as a service is composed of three important components; storage, servers and networks as is presented in Fig. 2. The network is responsible for interconnecting entire resources, servers can be any type of servers and storage that is attached to the servers. Amazon emerged as precursor of cloud computing since 2006 offering storage and basic processing via internet with Amazon Elastic Compute (Shankar, 2009) and Amazon Simple Storage Service (Palankar *et al*., 2008).

**Volunteer computing:** Volunteer computing is a type of distributed computing in which volunteers donate their own internet-connected computer resources (processing power, storage and Internet connection) to do one or more distributed computing projects (Sarmenta, 2001). Computers often employ only 10 to 20% of their total capability so there is a huge potential processing power available here in which by joining a hundred or million of them, it is possible to perform projects that essentially need huge processing power. The computing resources can be a desktop PC, laptops, mobile phones or tablets in the way that, by connecting these resources together, a single and super powerful computer is established that can do a large computational problems. Volunteer computing consists of two parts (Nov *et al*., 2010):

- **Computational aspects:** Related to problem of allocating and managing large computational jobs
- **Participation aspects:** Related to encouragement and persuasion tasks to attract more numbers of individuals to donate their computing resources to a project

There are lots of volunteer computing projects; SETI@home (Nov *et al*., 2010) which is a flagship volunteer computing project that was started in 1999 and since now has had over 3 milion volunteers, distributed.net, LHC@home and Rosetta@home. In volunteer computing projects, there is a big problem that need to carried out by a huge computational power, so the problem is divided into many executable tasks and each of them is solved by one or more donated nodes in parallel. Regarding to the parallel execution, it refers to a form of computation in which many computations are performed simultaneously based on the principle that large problems are divided into smaller ones which are then carried out in parallel (Kumar *et al*., 1994). Parallel computer programs are more difficult to write than sequential ones because parallel solution introduces many new types of potential software bugs. Communication and synchronization between the different subtasks are typically some of the greatest obstacles to have good parallel program performance (Patterson and Hennessy, 2008).

**VC middleware:** In order to compute scientific problems in volunteer computing, middleware should be assembled to handle computation tasks. The main goal of these platforms is to split the job and distribute the parts (tasks) all around the world and collects the results when ready. Middleware involves various elements (servers, networks, volunteers PCs, storage) to perform the jobs distributively. Actually volunteer computing middleware acts like an interface between OS and application software that need performance resources.

In order to design volunteer computing platforms some challenges and technical issues need to be addressed. Due to the nature of volunteer computing which is a wide distributed public computing, in the one hand users who own the resources are in a wide range of technical knowledge. In other hand users might own different resource architecture and specifications for instance different OSes and different software installed on that. From the user perspective, to attract and involve more users and consequently more computing resources the platform from user view should be ease of use which are including user interface design and abstracting the difficulties. From diversity of resources provided by donors, platform independence should be taken into consideration which means Macintosh or Linux owners can participate into the project as well as Windows users.

Security is the main challenge in volunteer computing as entire computing jobs are taken place in volunteers resources so the projects should be reliable since they are migrated into donated systems and performed on their host systems. Basically, the scheduler should guarantee the security concerns from user views in order to encourage the volunteers donating their resources.

The other important obstacle is application portability. Ideally, project owners who are willing to employ volunteer's resources must not be involved in the variety of hardware and platforms to perform their applications. As developing parallel application is not easy and maintaining and programming multiple versions of application for each platform might discourage the scientist to use volunteer computing platform.

Scalability and adaptability might become a hurdle since the project that is powered by volunteer computing platform, is performed by a large number of volunteers distributed in all around the world and resources provided by these users are in the risk of volatility and failures as public and free computing resources.

There are many volunteer computing frameworks that have been developed such as BOINC (Anderson *et al*., 2002), Xtremweb (Fedak *et al*., 2001) and HTCondor. In this section the overview on these volunteer computing middleware are described.

**BOINC:** The Berkeley Open Infrastructure for Net-work Computing (BOINC) (Anderson, 2004) is the most well- known framework for volunteer computing that is designed in a client-server approach. The BOINC client can be used to connect to different BOINC servers and participating in different projects. The BOINC project was established to power SETI@ home (Anderson *et al*., 2002) project. Then after the BOINC team decided to publish their work as an open-source middleware to allow the other projects which need free computing power, use the BOINC as computing platform. The BOINC aims to employ different range of computing resources from dedicated clusters to a low performance desktop computer. The BOINC client that is installed on volunteer's machine, is responsible to perform the jobs and the BOINC server has a manager and coordinator role in the system. The client application periodically seeks for any job available on the server and if it finds, downloads the workload and returns the result back to the server after computation. Each project requires a separate assembled server but volunteers can participate in multiple projects through a BOINC client. BOINC is supported by approximately 2,700,000 users that holds about petaflops (flops means floating point operations per second. Based on top 500 (Meuer, 2013) by comparing the power of BOINC with the most powerful supercomputer in the world, it is possible to argue that BOINC can be placed in the sixth place of the most powerful supercomputers in the world. The BOINC project relies on the virtual credit system that gives users the amount of virtual credit commensurate to the computing that they have done which may be network transfer or disk storage as well as computation (BOINC Credit, 2013). The BOINC strategy for harnessing of volunteer resources is that when volunteer machines become idle, BOINC starts computing usually with showing screen-saver.

From the server-side view, duties are based on several and distinctive tasks[1] and daemons[2]. The BOINC server infrastructure is designed to run on many unix-like OSes.

The BOINC core client is a daemon that manages com-munication with BOINC server, download of applications and workunits, client-side scheduling and finally upload of results. There is another program named BOINC manager that provides a graphical user interface of the BOINC client. There are many versions of the BOINC client designed for different platform; Windows, Apple Max OS X and Linux available on BOINC project webpage[3].

**BOINC server architecture:** The BOINC server is in charge of distributing tasks, collecting and storing successfully completed jobs from distributed clients. After a user joins the BOINC project, the server will manage the user registration process and records the machine specification that user holds in order to allocate the correct executing process for his architecture. After asking jobs by client, the application dispatches into the client machines and the execution process starts and the results are sent to the server to validate and store. The BOINC server is a collection of several components, each is responsible for different tasks. BOINC projects are identified by unmatched project URI. The project manager is in charge of configuring the application and also adds its workunits which are fetched and performed by the BOINC client and the processed results are returned to the server. The main and important component of BOINC is a MYSQL data storage that all information regarding registered users all information about the application and its workunits as well as writing the new information are the main duties of this data store. Every project powered by BOINC has the web page interface that aims to provide the communication with users. It provides the server statistics as well as some information about BOINC client installation, creating an account and participating into the project.

The rest of the server components consist of several daemons with different functionalities. These daemons and the communication between them are shown in Fig. 3. The rest part of this section describes these daemons (Korpela, 2012; Anderson, 2004).

**Scheduler:** The scheduler acts as a manager of requests that are dispatched by the BOINC clients. It runs at the background of BOINC project webpages and waits for requests form users. Requests can be a description of the host or asking for additional work or a list of completed jobs. Moreover, scheduler defines the limitation of disk used space and also the maximum computing time for each workunit. The reply of request consists of instances list and their associated jobs and the corresponding download URLs for the required files.

**Feeder:** The feeder is an auxiliary daemon that simplifies accessibility of database from the scheduler. In fact, the feeder fetches the workunits form the BOINC database into the memory that is shared between the feeder and the scheduler. This shared memory contains complete information regarding to the applications such as a platforms and version of applications. The scheduler scans this memory periodically to find jobs that likely send to available clients.

**Validator:** The validator is used for taking workunit results and check them if they are correct by comparing the instances of a job. If the comparison result is successful then it will become canonical representing the correct output. The validator also is responsible to grant

[1] Task is a program run at defined intervals (Anderson *et al.*, 2002)
[2] Daemon is a program that runs continuously (Anderson *et al.*, 2002)
[3] http://boinc.berkeley.edu

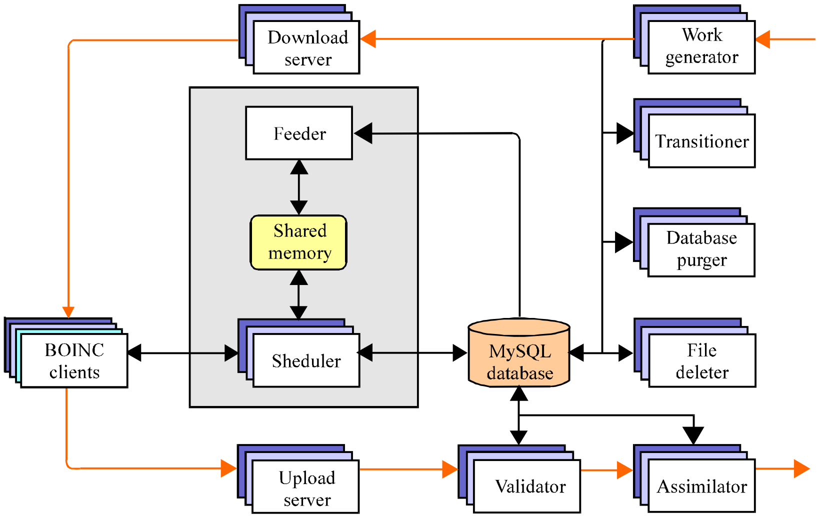


Fig. 3: Architecture of BOINC (Korpela, 2012)

credit to users and host that have returned the correct results and updates the associated database records.

**Assimilator:** The assimilator manages jobs that are completed including jobs that have a canonical instance or jobs with errors. Sometimes assimiliator create new wokunits based on completed results. The assimilator is in charge of writing outputs of successfully completed jobs into the database and archiving the output results.

**Deleter:** This daemon deletes information that become obsolete from the data servers. This daemon removes the workunits and also their input and output files that were marked as completed jobs.

**Transitioner:** The transitioner is responsible for handling different states that may occurred for available jobs. Depending on the situation, the transitioner may create new instances of job, mark the job as successfully done or performed with errors or enable validation or assimilation of the job.

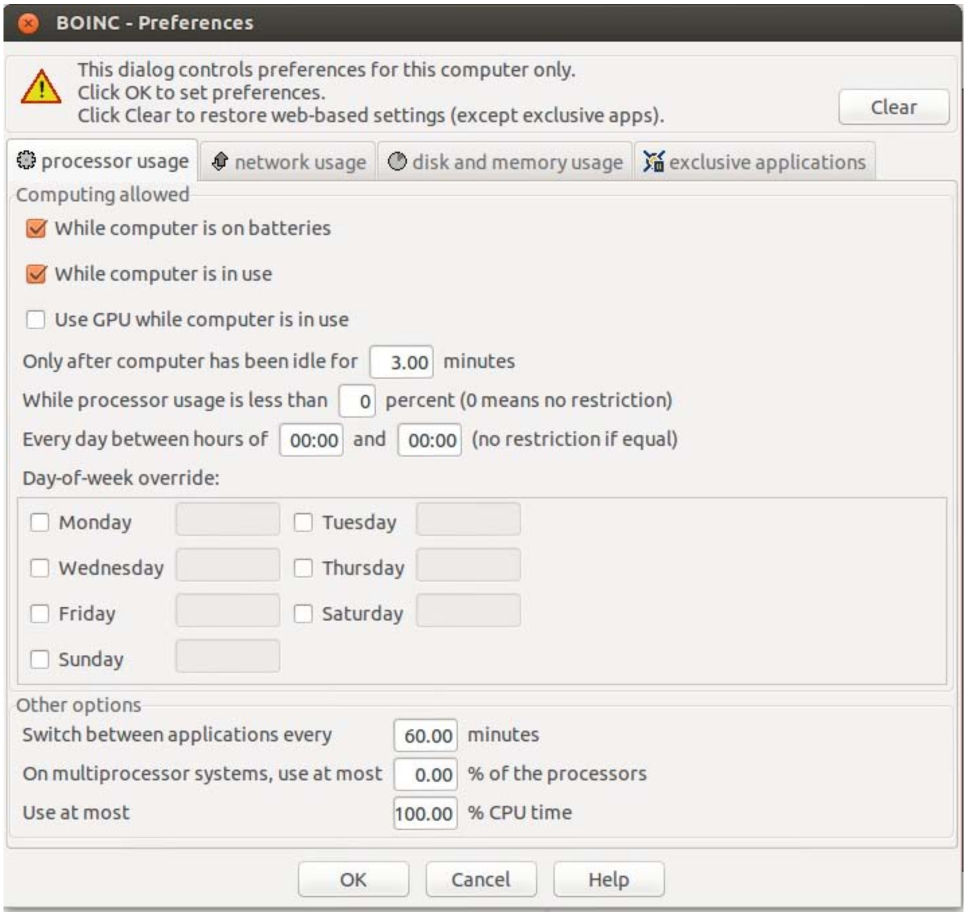
**BOINC client architecture:** The BOINC client package is downloaded by users. Volunteers that want to participate into the BOINC project should first register to obtain the unique identifier authentication which allows the BOINC project server to distinguish users for further processes. It is possible for a single client to attach to several different BOINC projects. The BOINC client is divided in four parts: The core client, the boinccmd component, the BOINC manager and the screansaver.

The BOINC core client is in responsible for handling communications with the BOINC server to do all computation related tasks. The boinccmd is used to control the BOINC core client via the command line. The BOINC manager provides the GUI representation of boinccmd allowing users to simply control BOINC client functionalities. Screansaver component represents graphics for the running application if it is provided by the project provider.

After the registration, user should attach into the BOINC project via uniquely project URL . When attached, the client tries to connect to the server scheduler and requests available tasks based on the user machine specification and also user preferences. The user can set some preferences option such as when the computation takes action or how many CPU cores to be used or also some limitation setting like maximum disk usage or network usage. Figure 4 shows the BOINC preference window.

If there is a work available then the BOINC client gets a URL for the workunits and computation process will start. After finishing of the computation, the result is uploaded to the server.

**Xtremweb:** Xtremweb is a multi-platform volunteer computing platform consists of three components, Client, Coordinator and Workers (Fedak *et al*., 2001). The workers send a message to server to get jobs and the server sends related files and also sends the application if it is not already existed in the workers machine. After finishing the execution process, worker contacts the



BOINC - Preferences

This dialog controls preferences for this computer only.
Click OK to set preferences.
Click Clear to restore web-based settings (except exclusive apps).

Computing allowed

☑ While computer is on batteries

☑ While computer is in use

□ Use GPU while computer is in use

Only after computer has been idle for 3.00 minutes

While processor usage is less than 0 percent (0 means no restriction)

Every day between hours of 00:00 and 00:00 (no restriction if equal)

Day-of-week override:

□ Monday □ Tuesday

□ Wednesday □ Thursday

□ Friday □ Saturday

□ Sunday

Other options

Switch between applications every 60.00 minutes

On multiprocessor systems, use at most 0.00 % of the processors

Use at most 100.00 % CPU time

Fig. 4: BOINC manager preferences settings

coordinator to send the results back to the server. In comparison with BOINC the architecture of both are almost the same with the central server and many workers/clients which are responsible for performing the jobs by pulling from the server. However, the most conspicuous feature of xtremweb in contrast with BOINC is that some users possess the right to submit new jobs to be executed by the rest workers on the network grid. The other stands out feature of xtremweb is the ability to not follow the centralized architecture in the way that workers might be able to send the results directly to client (XtremWeb Team, 2013).

**HTCondor:** Htcondor is a powerful resource management for workstation environment that is based on remote unix (Litzkow, 1987). HTCondor provides a job queueing mechanism, scheduling policy, priority scheme, resource monitoring and resource management (HTCondor, 2013). It is possible to use HTcondor as a cluster management of dedicated compute nodes. HTcondor provides transparently job checkpointing and migrating from user's view. In contrast with BOINC and xtremweb, HTcondor has a push model for its task distribution since the workers in condor grid network trust applications and so the jobs will be pushed from server to workers. HTcondor is composed of four different machines each serves more than one job in the same time. Central machine which exists only one per each condor pool, is responsible for collecting information and also acts like an interface between resources and resource requests. The other machines running under the same pool will contact the central machine and send their update status over the network. Execution machine is another part of condor architecture that is in charge of executing the job and provides the resources (CPU speed, memory and swap space) of condor pool. The next machine is the submit machine that it can be any one in the condor pool (including the central manager). Note that the resource requirements for this machine should be much more than the execution machine. Submit machine is in charge of saving the checkpoints of the jobs and also all system calls performed as remote procedure calls back to the submit machine. Moreover, binaries of jobs that are submitted to the execute machines are saved on the submit machine.

## DISCUSSION

In this section, we are are highlighting two important notions. First, what are the opportunities of utilising novel technologies in related to DG environments? Second, what are the open issues in current development of DGs applications and infrastructures?

**Opportunity:** Virtualization technology defines as a software abstraction layer between the hardware and the OS. With the advent of virtualization, it has been using in many computing infrastructure to enhance the infrastructure functionalities. Moreover, by enabling virtualization in volunteer computing middleware it is possible to incept a new computing paradigm namely Cloud$^{@}$home. In this section we provide the explanation of virtualization as well as introducing Cloud$^{@}$home notion.

**Virtualization:** Virtualization (Sahoo *et al*., 2010) is commonly defined as a technology that introduces a software abstraction layer between the hardware and the operating system and applications running on top of it. This abstraction layer is called Virtual Machine Monitor (VMM) or hypervisor which hides the physical resources of the computing system from the Operating System (OS).

By using virtualization in a grid environment it is possible to improve security due to isolation layer, enhance resource customization, boost controlling and utilization of resources and support legacy applications (Figueiredo *et al*., 2003). By enabling virtualization, utilising of multiple and different OSes on the same physical hardware in parallel is provided. Technically, the hardware of the system is tracked into distinct logical units named virtual machines (VMS).

**Cloud$^{@}$home:** The Vision and Issues: Cloud computing is the service oriented computing that offers the services by using virtualization. The other distinctive idea regarding to the virtualization is volunteer cloud computing or cloud$^{@}$home (Distefano *et al*., 2010). By building cloud-like-infrastructure from volunteer computing resources, new computing paradigm named volunteer clouds is established. This new powerful computing infrastructure returns the control of resources from commercial companies to users who can make decisions which and how much resources needed to be used in geographically distributed manner (Aversa *et al*., 2011). The idea is based on enabling virtualization technology in volunteer computing resources. In Table 1 and 2, we provided the review on works of applying virtualiztaion in VC frameworks by title, approaches, advantages of the approach and also the hypervisor which is used.

This type of cloud infrastructure addresses the problem of interoperability which is exists in cloud computing and also builds a customizable computing infrastructure in a lower scale and also in an affordable manner.

To achieve the volunteer cloud system goals there are many obstacles need to be addressed. One of the

Table 1: Enabling virtualization in VC

| Author/date | Title | Approach |
|---|---|---|
| Marosi *et al*. (2010) | Sandboxing for desktop grids using virtualization | They have developed a wrapper for launching a virtual machine and manage a task inside this virtual machine.<br>Communication direction is always from the wrapper to the task |
| Sanchez *et al*. (2011) | Volunteer clouds and citizen cyberscience for LHC physics | Use CernVM-FS to deploy the files of the experiment into the virtual machine |
| Ferreira *et al*. (2011) | Libboincexec: A generic virtualization approach for the BOINC middleware | Support the implementation behind an execution environment as a plug-in<br>Use VMM wrapper to integrate with BOINC |
| Marosi *et al*. (2012) | Towards a volunteer cloud system | The Generic BOINC Application Client (GBAC) is a virtualized based wrapper.<br>Instead of porting and registering applications individually, GBAC is ported and registered in BOINC |
| [1]BOINC (2012) | Vboxwrapper | Virtualization enables with the application named vboxwrapper<br>Input files are ported into the shared folder between host OS and guest OS |
| McGilvary *et al*. (2013) | V-BOINC: The virtualization of BOINC | Same with Vboxwrapper but virtual machine image will be downloaded automatically to the host.<br>This approach aims to use by application with dependencies |

[1] http://boinc.berkeley.edu/trac/wiki/VboxApps

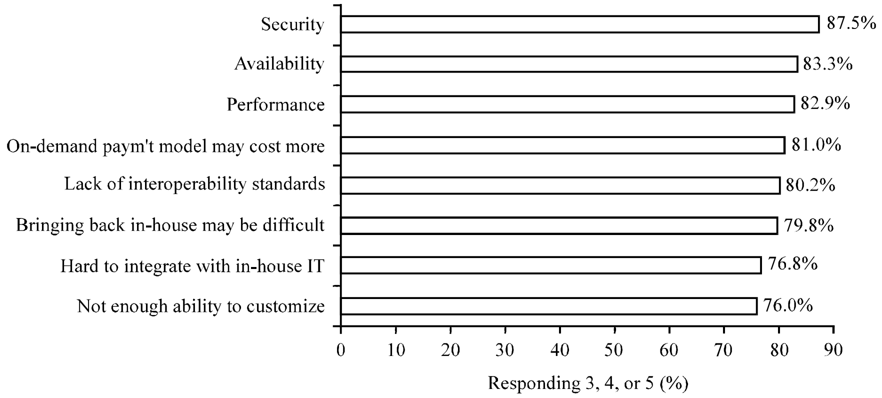


Fig. 5: Critical obstacles of cloud computing by IDC

Table 2: Enabling virtualization in VC

| Author/Date | Advantages | Hypervisor |
|---|---|---|
| Marosi *et al.* (2010) | Wrapper is intended to be portable across different volunteer computing infrastructure | QEMU |
| Sanchez *et al.* (2011) | Virtual appliance contains a complete job scheduler that performs job fetching from an external server by itself | VirtualBox |
| Ferreria *et al.* (2011) | Apart from the other part of library depends on BOINC | VirtualBox/VMware |
| Marosi *et al.* (2012) | Providing virtualized environment for various distributed computing infrastructures (DCIs)<br>Does not rely on any middleware specific functionalities, thus it is possible to use it on any DCIs that are supported by DC-API | VirtualBox |
| [1]BOINC (2012) | No modification need in BOINC core client | VirtualBox |
| McGilvary *et al.* (2013) | This approach gets applications with dependencies | VirtualBox |

biggest challenges of moving from costly and modern cloud's datacenters to volunteer's resources is the volatility and availability of resources. In volunteer computing it is common that computing machines are got out of the project by either resource's owners or some technical occurrences (e.g., system crashing or power problem). Consequently fault-tolerance has considered as an important effort to establish volunteer cloud. Another important problem that need to be addressed is convincing volunteers to donate their resources in a different level of accessibilities than before as resources are not used only for scientific problems but also used by commercial providers.

**Open issues:** In this section, we provide current obstacles in development and adoption of cloud computing applications. Moreover, we highlight the most important challenges in volunteer computing frameworks.

**Challenging issues for cloud computing:** International Data Corporation (IDC) conducted a survey (Gens, 2009) (Fig. 5) and asked from 263 IT organizations to rank the critical obstacles that prevent cloud computing from being adopted. A glance at the Fig. 5 provided, reveals that security, availability and performance are the top three concerns by organizations. Without doubt, security has played an important role as a obstacle in cloud computing. Corporations and individuals are concerned about how security can be implemented in this new environment. Popovic and Hocenski (2010) highlighted security concerns related to cloud computing infrastructure in three views; Security issues and challenges view, Security management standards view and Security management models view.

Table 3: Comparing public cloud SLAs

| Cloud vendor | Name of services | Monthly uptime percentage |
|---|---|---|
| Google | Google apps | < 99.9% -> = 99.0% (Google, 2013) |
| Amazon | Amazon EC2 | < 99.5% -> = 99.0% (Amazon EC2, 2013) |
| Amazon | Amazon S3 | < 99.9% -> = 99.0% (Amazon S3, 2013) |
| Apple | iCloud | No-offered (Apple Inc., 2013) |
| Microsoft | Cloud services | <99.95% (Microsoft, 2013) |
| Dropbox | Dropbox | No-offered (Dropbox, 2012) |
| Cloud flare | CloudFlare | 100% (CloudFlare, 2013) |

From the availability point of view, organizations worry about availability of cloud computing services. Cloud vendors offer Service Level Agreements (SLAs) for the uptime and availability of services, while these availability SLAs should satisfy the needs of almost any cloud applications to convince enterprises and IT organizations to move into cloud. Table 3 provides a comparison availability SLAs of well-known cloud

[1] http://boinc.berkeley.edu/trac/wiki/VboxApps

Table 4: Cloud computing service outages in 2013 (Raphael, 2013; Choudhury, 2013; Staff, 2013)

| Cloud vendor | Error | Duration | Date |
|---|---|---|---|
| Twitter | Timelines came up blank and tweets went undelivered | Short time | January 3 |
| Dropbox | Client-syncing and file-uploading failure | Around 16 h | January 10 |
| Facebook | Friends' status updates failure | 2-3 h | January 28 |
| Amazon | Amazon home page goes down | 49 min | January 31 |
| Microsoft | Office 365 editing suite and Out-look.com mail service both stut-tered | Around 2 h | February 1-2 |
| Microsoft | Microsoft azure cloud suffered a worldwide service interruption | Almost a full day | February 22 |
| Cloud flare | company's own site and all of its services kicked the bucket, | About an hour | March 3 |
| Google | Slow load times or full-on timeouts on Google drive | About 17 h total | March 18-19 |
| Apple iCloud | Services mostly experienced failures in authentication | Several hours | April 3 |

providers. However, infrastructure downtime is inevitable due to many unexpected issues. Table 4 illustrates the recorded cloud computing service outages in 2013 along with the error and duration. Google provided an application named "Apps Status Dashboard" (Google Team, 2013) that provides for users to view current status of all Google services.

Cloud computing infrastructure takes a large expense re-garding datacenters deployment. Considering approximately $53 million each year only for servers or about $10 million for powering (Greenberg *et al*., 2009). Finding a proper solution to minimise the energy consumption of cloud infrastructure and consequently reducing maintenance costs of cloud computing caused to the new cloud area named "Green Cloud Computing" (Buyya *et al*., 2010). The other alternative solution is building cloud infrastructure from grid resources which named Clouds@home (Cunsolo *et al*., 2010). One concern that is arisen in using cloud based services is vendor lock- in which refers to the inability of moving resources from one cloud vendor to another (Armbrust *et al*., 2009). Vendor lock-in stems from the absence of interoperability features in current cloud computing environments (Armbrust *et al*., 2010). Interoperability is defined as the ability of moving workloads and data from one cloud vendor to another. There is a need to investigate common standards to enable interoperability and make available a freedom to import and export services between different vendors. The main challenges that need to be addressed in deploying interoperable cloud are the cost of bandwidth and also storage during data migration. Migration of VMs from one cloud service provider to another one has been considered as a key to enable interoperability in cloud computing. In other words, going with a cloud solution means buying into a specific protocols and standards of cloud provider and thus future migration becomes as a difficult and costly process. A possible solution to overcome the problem of interoperability is the usage of open standards. There are many standardization efforts like cloud-standards.org, NIST (NIST, 2011), SIENA, CMWG (CMWG, 2007), DMTF's Virtualization Management Initiative (DMTF, 2009) and others that aim to enable open and interoperable clouds.

## CHALLENGING ISSUES FOR VC

In this section, main challanging issues of existing VC systems are described (Choi *et al*., 2007; Rimal *et al*., 2009):

**Volatility:** In volunteer computing it is common that computing machines are got out of the project by either resource's owners or some technical occurrences (e.g., system crashing or power problem). A scheduler should support mechanisms to guarantee the availability of services.

**Lack of trust:** In Desktop Grid, processing tasks take place in owners computer therefore volunteers should trust the jobs and workloads. It is possible to see corrupted results because of malicious interposition. A scheduler should take some actions and procedure to guarantee the correctness of results.

**Failure:** Resource provider of volunteer computing systems are connected through the internet and it is so often they are disconnected because of crashing or link failures. A scheduler should tolerate the failures and volatility.

**Heterogeneity:** The resource provided by volunteers are in wide range of specifications and properties such as CPU, memory, network bandwidth and failure rate. These considerations cause the overall performance and make delay and so decision making looks difficult for the scheduler.

**Voluntary participation:** The resources provided by volunteers are free to join and leave even in execution time. In the one hand, finding proper mechanisms to attract more users should be taken into consideration. In the other hand in order to have a long donating time by donors, a scheduler should take some actions on rewarding mechanism for resource providers.

## CONCLUSION

Computing paradigm has emerged as a solution to address complex computers or mathematics related issues.

In the reflection of remarkable advantages of cloud computing technology such as pay-per-use pricing model, scalability of services and delivering hosted services over the internet, the number of applications and services hosted in the cloud increases substantially. In this study, the advantages and disadvantages of cloud computing as well as cloud computing services and models have been highlighted. Volunteer computing as a powerful computing prototype has been explained. The issues of volunteer computing systems were introduced. Clouds@home as a new computing paradigm was introduced and the issues and the future forecast of this computing notion were highlighted.

## ACKNOWLEDGMENTS

We wish to thank Dr. Rodrigo N. Calheiros, a Research Fellow in the Department of Computing and Information Systems, The University of Melbourne, Australia , for his support, encouragement and insightful comments on this research article. This work is supported by the Faculty of Computing, Universiti Teknologi Malaysia under Grant No. R.J130000.7828.4L108.

## REFERENCES

Amazon EC2, 2013. Amazon EC2 service level agreement. http://aws.amazon.com/ec2/sla/.

Amazon S3, 2013. Amazon s3 service level agreement. http://aws.amazon.com/s3/sla/.

Anderson, D.P., J. Cobb, E. Korpela, M. Lebofsky and D. Werthimer, 2002. SETI@home: An experiment in public-resource computing. Commun. ACM., 45: 56-61.

Anderson, D.P., 2004. BOINC: A system for public-resource computing and storage. Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing, November 8, 2004, Pittsburgh, USA., pp: 4-10.

Anderson, D.P., E. Korpela and R. Walton, 2005. High-performance task distribution for volunteer computing. Proceedings of the 1st International Conference on e-Science and Grid Computing, July 1, 2005, Melbourne, Vic., Australia, pp: 196-203.

Apple Inc., 2013. iCloud terms and conditions. https://www.apple.com/legal/internet-services/icloud/en/terms.html.

Armbrust, M., A. Fox, R. Griffith, A.D. Joseph and R.H. Katz *et al.*, 2009. Above the clouds: A Berkeley view of cloud. Technical Report No. UCB/EECS-2009-8. http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.pdf.

Armbrust, M., A. Fox, R. Griffith, A.D. Joseph and R. Katz *et al.*, 2010. A view of cloud computing. Commun. ACM, 53: 50-58.

Aversa, R., M. Avvenuti, A. Cuomo, B. Di Martino and G. Di Modica *et al.*, 2011. The Cloud@Home Project: Towards a New Enhanced Computing Paradigm. In: Euro-Par 2010 Parallel Processing Workshops, Guarracino, M.R., F. Vivien, J.L. Traff, M. Cannatoro and M. Danelutto *et al.* (Eds.). Springer, New York, ISBN-13: 9783642218781, pp: 555-562.

BOINC Credit, 2013. Computation credit. July 2013. http://boinc.berkeley.edu/wiki/Computation_credit.

Berman, F., G. Fox and A.J.G. Hey, 2003. Grid Computing: Making the Global Infrastructure a Reality. John Wiley and Sons, London, UK., ISBN-13: 9780470853191, Pages: 1012.

Buyya, R., 1999. High Performance Cluster Computing: Programming and Applications. Prentice Hall, New Jersey, USA., ISBN-13: 9780130137852, Pages: 664.

Buyya, R., A. Belonglazov and J. Abawajy, 2010. Energy-efficient management of data center resources for cloud computing: A vision, architectural elements and open challenges. Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications, July 12-15, 2010, Las Vegas, NV., USA., pp: 6-20.

CMWG, 2007. Distributed management task force: Cloud. Cloud Management Working Group (CMWG). http://www.dmtf.org/standards/cloud.

CRN Staff, 2013. The 10 biggest cloud outages of 2013 (so far). http://www.crn.com/slide-shows/cloud/240158298/the-10-biggest-cloud-outages-of-2013-so-far.htm?pgno=11.

Choi, S., H.S. Kim, E.J. Byun, M.S. Baik, S.S. Kim, C.Y. Park and C.S. Hwang, 2007. Characterizing and classifying desktop grid. Proceedings of the 7th IEEE International Symposium on Cluster Computing and the Grid, May 14-17, 2007, Rio de Janeiro, Brazil, pp: 743-748.

Choudhury, A., 2013. Google outage. August 19, 2013. http://webmasterfacts.com/google-outage/.

CloudFlare, 2013. Business service level agreement. http://www.cloudflare.com/business-sla.

Cunsolo, V.D., S. Distefano, A. Puliafito and M. Scarpa, 2010. Open and Interoperable Clouds: The Cloud@Home Way. In: Cloud Computing: Principles, Systems and Applications, Antonopoulos, N. and L. Gillam (Eds.). Springer, New York, ISBN-13: 9781849962414, pp: 93-111.

DMTF, 2009. Open virtualization format. http://www.dmtf.org/standards/ovf.

Distefano, S., V.D. Cunsolo and A. Puliafito, 2010. A taxonomic specification of cloud@home. Proceedings of the 6th International Conference on Intelligent Computing, August 18-21, 2010, Changsha, China, pp: 527-534.

Dropbox, 2012. Dropbox privacy policy. March 2012. https://www.dropbox.com/privacy.

Fedak, G., C. Germain, V. Neri and F. Cappello, 2001. XtremWeb: A generic global computing system. Proceedings of the 1st IEEE/ACM International Symposium on Cluster Computing and the Grid, May 15-18, 2001, Brisbane, Qld., Australia, pp: 582-587.

Ferreira, D., F. Araujo and P. Domingues, 2011. libboincexec: A generic virtualization approach for the BOINC middleware. Proceedings of the IEEE International Symposium on Parallel and Distributed Processing Workshops and PhD Forum, May 16-20, 2011, Shanghai, China, pp: 1903-1908.

Figueiredo, R.J., P.A. Dinda and J.A.B. Fortes, 2003. A case for grid computing on virtual machines. Proceedings of the 23rd International Conference on Distributed Computing Systems, May 19-22, 2003, Providence, RI., USA., pp: 550-559.

Foster, I. and C. Kesselman, 2004. The Grid 2: Blueprint for a New Computing Infrastructure. 2nd Edn., Morgan Kaufmann Publishers Inc., San Francisco, CA., USA., ISBN-13: 9781558609334, Pages: 748.

Foster, I., Y. Zhao, I. Raicu and S. Lu, 2008. Cloud computing and grid computing 360-degree compared. Proceedings of the Grid Computing Environments Workshop, November 12-16, 2008, Austin, TX., USA., pp: 1-10.

Gens, F., 2009. New IDC IT cloud services survey: Top benefits and challenges. December 15, 2009. http://blogs.idc.com/ie/?p=730.

Google Team, 2013. Apps status dashboard. http://www.google.com/appsstatus#hl=en&v=status.

Google, 2013. Google apps service level agreement. November 2013. http://www.google.com/apps/intl/en/terms/sla.html.

Greenberg, A., J. Hamilton, D.A. Maltz and P. Patel, 2009. The cost of a cloud: Research problems in data center networks. ACM SIGCOMM Comput. Commun. Rev., 39: 68-73.

HTCondor, 2013. What is HTCondor? http://research.cs.wisc.edu/htcondor/description.html.

Huhns, M.N. and M.P. Singh, 2005. Service-oriented computing: Key concepts and principles. IEEE J. Internet Comput., 9: 75-81.

Korpela, E.J., 2012. SETI@home, BOINC and volunteer distributed computing. Annu. Rev. Earth Planet. Sci., 40: 69-87.

Kumar, V., A. Grama, A. Gupta and G. Karypis, 1994. Introduction to Parallel Computing: Design and Analysis of Algorithms. Benjamin/Cummings Publishing Co., IPC., North-Holland, ISBN: 9780805331707, Pages: 597.

Litzkow, M.J., 1987. Remote UNIX: Turning idle workstations into cycle servers. Proceedings of the Summer USENIX Conference, June 8-12, 1987, Phoenix, AZ., USA., pp: 381-384.

Marosi, A.C., P. Kacsuk, G. Fedak and O. Lodygensky, 2010. Sandboxing for desktop grids using virtualization. Proceedings of the 18th Euromicro International Conference on Parallel, Distributed and Network-based Processing, February 17-19, 2010, Pisa, Italy, pp: 559-566.

Marosi, A., J. Kovacs and P. Kacsuk, 2012. Towards a volunteer cloud system. Future Gener. Comput. Syst., 29: 1442-1451.

McGilvary, G.A., A. Barker, A. Lloyd and M. Atkinson, 2013. V-BOINC: The virtualization of BOINC. Proceedings of the 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, May 13-16, 2013, Delft, Netherlands, pp: 285-293.

Mell, P. and T. Grance, 2011. The NIST definition of cloud computing. NIST Special Publication No. 800-145, National Institute of Standard and Technology, U.S. Department of Commerce, September 2011.

Meuer, H., 2013. Top 500 supercomputer sites. http://www.top500.org/.

Microsoft, 2013. Service level agreements. http://azure.microsoft.com/en-us/support/legal/sla/.

NIST, 2011. NIST cloud computing program. http://www.nist.gov/itl/cloud/.

Nov, O., D. Anderson and O. Arazy, 2010. Volunteer computing: A model of the factors determining contribution to community-based scientific research. Proceedings of the 19th International Conference on World Wide Web, April 26-30, 2010, Raleigh, NC., USA., pp: 741-750.

Palankar, M.R., A. Iamnitchi, M. Ripeanu and S. Garfinkel, 2008. Amazon s3 for science grids: A viable solution? Proceedings of the 2008 International Workshop on Data-Aware Distributed Computing, June 23-27, 2008, Boston, MA., USA., pp: 55-64.

Patterson, D.A. and J.L. Hennessy, 2008. Computer Organization and Design: The Hardware/Software Interface. 4th Edn., Morgan Kaufmann Publishers Inc., San Francisco, CA., USA., ISBN-13: 978-0123744937, Pages: 912.

Popovic, K. and Z. Hocenski, 2010. Cloud computing security issues and challenges. Proceedings of the 33rd International Convention on Information and Communication Technology, Electronics and Microelectronics, May 24-28, Opatija, Croatia, pp: 344-349.

Raphael, J.R., 2013. The worst cloud outages of 2013 (so far). http://www.jrstart.com/4443/the-worst-cloud-outages-of-2013-so-far/.

Rimal, B.P., E. Choi and I. Lumb, 2009. A taxonomy and survey of cloud computing systems. Proceedings of the 5th International Joint Conference on INC, IMS and IDC, August 25-27, 2009, Seoul, South Korea, pp: 44-51.

Sahoo, J., S. Mohapatra and R. Lath, 2010. Virtualization: A survey on concepts, taxonomy and associated security issues. Proceedings of the 2nd International Conference on Computer and Network Technology, April 23-25, 2010, Bangkok, Thailand, pp: 222-226.

Sanchez, C.A., J. Blomer, P. Buncic, G. Chen and J. Ellis *et al*., 2011. Volunteer clouds and citizen cyberscience for LHC physics. J. Phys.: Conf. Ser., Vol. 331. 10.1088/1742-6596/331/6/062022

Sarmenta, L.F.G., 2001. Volunteer computing. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA., USA.

Shankar, S., 2009. Amazon elastic compute cloud. http://www.cs.berkeley.edu/~volkov/cs267.sp09/hw0/results/Sushant.Shankar.pdf.

Tsai, W.T., X. Sun and J. Balasooriya, 2010. Service-oriented cloud computing architecture. Proceedings of the 7th International Conference on Information Technology: New Generations, April 12-14, 2010, Las Vegas, NV., USA., pp: 684-689.

Vaquero, L.M., L. Rodero-Merino, J. Caceres and M. Lindner, 2009. A break in the clouds: Towards a cloud definition. ACM SIGCOMM Comput. Commun. Rev., 39: 50-55.

XtremWeb Team, 2013. Introduction to XtremWeb. http://www.xtremweb.net/introduction.html.