Journal of Theoretical and Applied Information Technology
31st May 2020. Vol.98. No 10

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

# A REVIEW OF DESKTOP GRID COMPUTING MIDDLEWARES ON NON-DEDICATED RESOURCES

[1] MAHATHIR RAHMANY, [2]ELANKOVAN A. SUNDARARAJAN, [3]ABDULLAH MOHD ZIN
[1,2,3]Fakulti Teknologi Sains dan Maklumat, Universiti Kebangsaan Malaysia (UKM), Bangi, Selangor, 43600, Malaysia
E-mail: [1]mahathir@siswa.ukm.edu.my, [2]elan@ukm.edu.my, [3]amzftsm@ukm.edu.my

**ABSTRACT**

Grid Computing is a distributed computing technology that can provide users with powerful computing and storage capacity. Most of the applications of grid computing are now replaced by the use of cloud computing. However, there is an area where the grid computing technology can still be utilized, that is in the area of desktop grid computing (DGC). With the growth of desktop computer belonging to an organization and the increasing of internet penetration worldwide, desktop grid computing can be a good alternative to provide the cheapest high-power computing to solve computing intensive applications by utilizing non-dedicated resources. DGC can also be utilized if an application involved some sensitive and secured data. Another area where DGC can be useful is to support the concept of volunteer computing which is expected be more popular in the future since devices that can be volunteered are not only limited to desktop computers but also other computing devices such as smart phones. The main objective of this paper is to review some available middlewares for DGC. This paper also discusses the future direction of DGC.

**Keywords:** *Desktop Grid Computing; Cloud Computing, Middleware, Volunteer Computing.*

## 1. INTRODUCTION

Parallel computing is a type of computation that allows the execution of processes to be carried out simultaneously [1]. Figure 1 shows the evolution of Parallel computing to Cloud computing.

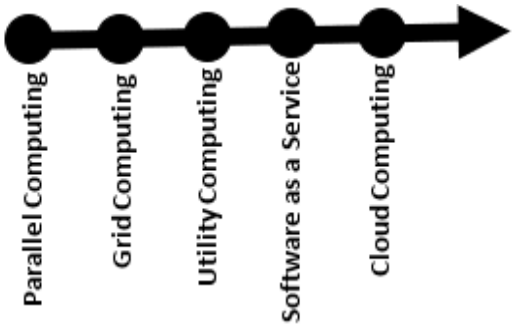


*Figure 1: Cloud computing evolution* [2], [3]

The concept of Grid Computing (GC) is explained briefly by [4] as integrated system which contain servers, storage on the networks in order to get significant spike of computing power and storage capacity. Grid computing are used for computing intensive application such as Bioinformatics, Cheminfor-matics, Medical Informatics, Physical Simulations, Compute-Intensive Analysis of Large Data, and Biology-Inspired Algorithm [5].

Even though GC offer numbers of computing resources but it also come up with intricate problem, such as profoundly difficult on hardware scaled up or down [6]. In the same time Cloud Computing (CC) which is a new model on employing technology [7], [8] comes to complete GC in terms of previous problem [9]. CC has been used by researcher to enhance their research productivity [10]. However, there is an area where the grid computing technology can still be utilized, that is in the area of desktop grid computing (DGC). And now DGC has become prominent options of CC [11] and well-known tool in solving scientific computation [12]. CC shares the basic concept as DGC where to harness the unused of computer computing resource [11]. Components of DGC consists of non-dedicated resources such as desktop or laptop computers [13] connected to a network [12]. DGC is a cheaper solution to obtain a high computing power in order to solve computing intensive applications [14]. The taxonomy of Desktop Grids in shown in Figure 2 [15].

The main strength of DGC is the steadily increasing availability of desktop computers [16].

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

DGC runs with two main pillars [17], which are computational and participative pillars. Computational pillar refers to managing and setting up DGC, so it can be utilized as efficiently as possible. Participative pillar intends to attract as many volunteers as possible.

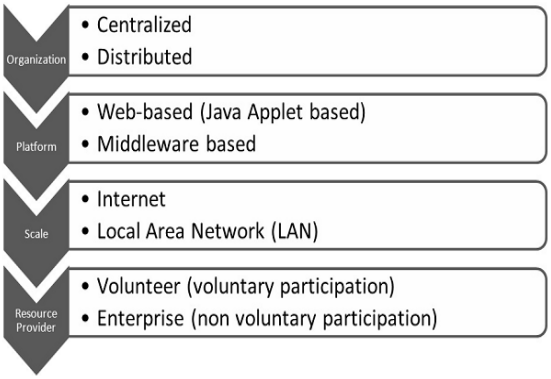


*Figure 2: A Taxonomy of Desktop Grid* [15]

Another advantage of DGC is that its middleware is easy to be installed [18]. The middleware used by DGC is different from the one used by grid computing. DGC requires middleware that can address the issues in very dynamic environment and situation, such as unstable connection, varying bandwidth or machines heterogeneous operating systems and architecture, and anonymous volunteers.

The structure of this paper is as follows. Section 2 talks about the motivation and aim of the article. Some of DGC middlewares are discussed in section 4. Section 5 is the discussion and finally, section 6 concludes this paper.

## 2. THE MOTIVATION AND AIM

The main concept of CC is to shared computing resource to maximize the proficiency with minimum price [19], [20]. In the same time as have mentioned above, DGC shared the same concept as CC. So, the motivation of this research is to give insight the difference between CC and DGC and finally can demonstrate how in some ways DGC can outperform CC.

## 3. MIDDLEWARE

Inherently, middleware is related to message delivery [21]. It means middleware is a bridge to manage the various information between client and server. The concept of middleware appeared initially in 1980s where for the first time researchers develop primary component for midleware such as remote procedure call, file service, and directory service [22]. Sadjadi [23] states that middleware is a software associate to wrap up the various services that are among network operating system and user application. He also added that the middleware is very useful to overcome complicated repetitive work on code writing for communication process between client and server across platform.

The main functions of middleware are:

- ✓ To synchronize among application programs, lower-level hardware, and software infrastructure that make the connection and interoperation of any application is easy [24].
- ✓ To simplify integration process of any components in order to run application, lower-level hardware, or software infrastructure [24].
- ✓ Middleware avoids the rewriting of programs that are low-level, tedious, and error-prone; which eventually can save time and cost on software lifecycle [24].

There are two types of middleware [25]: integration middleware and application middleware. Integration middleware means that the middleware has its own adapting techniques to a completely different environment. Unlike integration middleware, application middleware has only limited techniques for what the application is made.

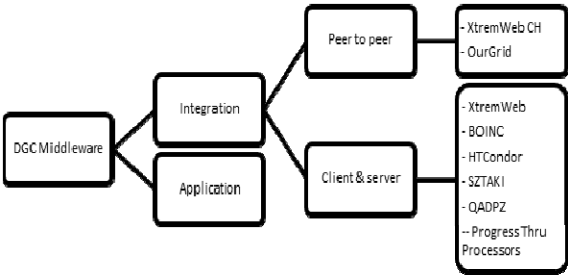


*Figure 3: DGC Middleware Taxonomy* [25]

## 4. DGC MIDDLEWARE

Many middlewares have been developed for desktop grid computing, such as BOINC [26], Strewed [26], OurGrid [26], SZTAKI Desktop Grid [26], and HTCondor [26]; however not all of them are active. Some middleware does not have ongoing research anymore. In this paper, we concentrate our discussion on the well-known middlewares. Based on the architecture, the above middleware can be separated into two parts, Peer-to-peer and Client & Server.

Some of the terms used in DGC middleware are:

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

**Server**. Server is a device that has the responsibility to allocate the work to each worker [27].

**Worker**. Worker is a device that runs the job received from the server voluntarily [27].

**Result Error**. Result Error is wrong result or also not in correct range that has been appointed [27]. The result error is caused by some factors such as sabotage, intervention by worker owner, malicious workers, etc.

**Error rate**. Error rate is proportion of error or incorrect result between the tolerate result after the computation is finished [27].

**Redundancy**. Redundancy is proportion of how many duplicate work that is given to worker with verified number N of work units [27].

**Error Detection**. Error Detection is how the middleware can verify or ascertain there are any error on the computation by recompute the already known result to other workers [27], [28].

**Spot Checking**. Spot Checking is to test the disordered workers with the job that has identify as the correct result. If among of the workers send dissimilar result as expected by the server, so the workers will be boycott for the next computation [27].

**Sabotage-Tolerance**. The technique to overcome the inconstancy of computation by applying the voting [29].

**Voting**. Voting is a way to overcome the Sabotage-Tolerance problem by asking some workers to do the same jobs many times. The most number of the same result that is generated by the workers is the correct result [30].

**Scheduler**. Scheduler has the responsibility to manage the data traffic between workers and server. It manages tasks to be executed by workers and control the result generated by workers to server [31].

**Checkpoint**. Checkpoint is saving the information where the node unsuccessful to work. So when the job is restarted it is no need to start from the first, it just can start from the checkpoint [32].

**Tasks**. A portion of jobs [33]. These tasks are distributed from server to clients.

**Jobs**. A group of task, to be executed in parallel [33].

**Network of Favors**. The right to use resources on network as equal as the resources that have been donated [34].

**Free riders**. Selfish user where only want to consume the resource without willing to provide his computing resources [35].

**Pull model**. A concept in DGC where the clients request job from the server [36].

**Push model**. A concept in DGC where the server pushes job to the clients [37].

There are two types of DGC middleware: peer-to-peer and client server. These middlewares are distinguished in term of organization, behavior and network structure [38] as shown in Table 1 below.

*Table 1: Client Server vs Peer to Peer* [38]

| | Client Server | Peer to Peer |
|---|---|---|
| Organization | Centralized | Decentralized |
| Behavior | Dependence | Autonomy |
| Network Topology | Structured | Unstructured |

### 3.1 Peer-to-Peer

Peer-to-peer architecture is very suitable to be implemented on DGC, because its architecture supports [39]: (i) Platform heterogeneity, (ii) Symmetric view, the node can be a server or a client; (iii) Natural scalability, the master will never overload because the node is a master and a client. Moreover, this technology also promising low-cost desktop grid by giving the possibility just harnessing smaller desktop grid [40].

#### 3.3.1 XtremWeb-CH [41]

One of DGC middleware based on peer-to-peer is XtremWeb-CH (XWCH). This middleware consists of four modules: coordinator, worker, warehouse and broker; and the coordinator as the main modules [42]. The function of coordinator is to coordinate between client as users and server as workers. Every node has its own worker to do the job and to supply the task among them. To make the application that is submitted by the user is understand by XtremWeb-CH, the broker has to alter the application into the format that can know by this middleware [39].

Worker will send the calculation result to the one or more warehouses. The warehouse pretends as repository or file server to keep data and binary executables. As policy in Xtremeweb-CH, the worker must submit the execution result at least to one warehouse [43]. To communicate to

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Coordinator, worker send four types of cue (1) WorkRegister, (2) WorkAlive, (3) WorkRequest, and (4) WorkResult [43].

#### 3.3.2 OurGrid

Another middleware based on this architecture is OurGrid. OurGrid promises the slight on Installing, configuring, and customizing [44]. To overcome the freeriding problem or someone who uses resources without re-donate the resource to the network [45], OurGrid implements the Network of Favors, means the resource that you can get from the network based on how many resources that you have donated. Also next time, if you have donated many resources; you will get the priority to get the resource from others [33].

There are three main components on OurGrid architecture, the OurGrid Community, MyGrid and SWAN [33]. MyGrid as agent to link between user and the server by providing tasks, jobs, and grid machines; OurGrid Community to construct the grid that will be used by MyGrid; and SWAN as controlling the grid process running on the secure way [33]. The application concept that is used by OurGrid is a bag-of-tasks [46] which it means each of those parallel applications is independent [44].

### 3.2 Client-Server

Client-Server has different architecture compare to Peer to Peer (p2p) architecture. In this model, there are one or more servers that have control over client, such as centralized security databases to control access to shared resources on servers. In this model also, server may in turn be clients of other servers [47]. Some of middlewares based on this architecture are XtremeWeb, BOINC, HTCONDOR, SZTAKI Local Desktop Grid, QADPZ and Progress Thru Processor.

#### 3.2.1 XtremWeb [48]

XtremWeb, which is written in Java, Perl and C, has two main structures architecture: Worker and Server. Worker has two jobs: (1) To prepare the available desktop grid resources to do the XtremWeb computation. The worker decides if the desktop grid resources is ready to be deployed by monitoring the user availability (working mouse or keyboard), recognizing the CPU, I/O and memory usage; and (2) To do the work which is commanded by the server. The worker will launch the thread when the desktop grid computer is ready to executes the job. This thread has some functions, such as: controlling, monitoring, computing and alive [48].

In other side the server architecture has three main modules: pool of applications, pool of jobs, accounting modules, and web user interface. Pool of Applications has the responsibility on application with multiparameter consecutive and assigning for pre-compiled binaries to be distributed to varying platform [48].

Besides those three main modules, server also has another two modules to support the server. They are Scheduling & Server Specialization and Implementation. Scheduling & Server Specialization has some components, (1) Dispatcher. The aim of this Dispatcher is transmitting the task from the task pool to the scheduler. (2) Scheduler. To ensure every task is done well. The policy that is used to accomplish the task is FIFO (first in first out). It means every task will be done sequentially based on which the first task is. For the distribution task, scheduler applies the pull model concept which means the worker asks scheduler for the task. The worker is determined by its run time environment and the existence of a pre-build binary of the application [48].

#### 3.2.2 BOINC (Berkeley Open Infrastructure for Network Computing) [49]

BOINC was introduced for the first time in 2002. Now it becomes ones of prominent middleware [16]. Not just for DGC middleware, this is also prepared for social network. It describes sharply from the features that is provided, for example it provides rewards such as a new cryptocurrency that is called Gridcoin [50], [51] for who has donate resources the most, provide the forum to help each other of volunteers; this is include also for the software developer. The last, it also provide for volunteers and scientist to take a part together to promote the science [40]. To bring about more computing resources, BOINC has provided the client application for Linux, Microsoft Windows, Apple Mac OS X [52], [53], and Android [53], [54] users.

BOINC, which adopted the pull model as task distribution, consists of two side, client and server. On client side, BOINC has three main components: (1) Client, (2) Manager, and (3) Screensaver [55]. To run any BOINC project, it must has its own server and every server consists of three parts: (1)Web interface; (2) task server; and (3) data server [56].

For running its task, BOINC has eight elements as in server side. Those elements are work generator, scheduler, feeder, transitioner,

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

assimilator, file deleter, and database purger [56]. Work generator publishes the new work unit that will be executed by the application; The feeder takes the job from the database and distributes to the queue; Than the transitioner shall control between the work unit and the result; Assimilator will save any valid result to the other database. The unused files and work unit will be removed from the database by the deleter and database purger [57].

The BOINC processes start from Work Unit Generator, where from this first step, Work Unit Generator has responsibility to produce new work unit to be sent to clients by putting them in queue through feeder. Before the work units are given to the client, they must obey the scheduler regulation. The scheduler is middle person on managing information in and out from clients [58]. To avoid any malicious action to the system, BOINC just give approval to the code that has digitally signed [18].

### 3.2.3 HTCONDOR [59]

HTCondor is the first middleware which utilize the idle resource [60]. It is belonging to HTCondor Research Project at the University of Wisconsin-Madison (UW-Madison). Basically, HTCondor consists with three parts: (1) Job Submitters; (2) Machine Owners; and (3) Pool administrator. Both could be the same person. The Job Submitters can request the specification of how the project work. Usually the specification is made based on the project. For example, if the project should only work under Linux machine, must be run on maximum memory. The request is prepared on the submission file and submitted when the project ready to be run.

Machine owner has right to control how the machine works. As an instance the machine run the HTCondor machine in the midnight only with the maximum memory can be used or when the machine is idle. The owner also has authorization to choose the type of project. Pool Administrator ties between Job Submitters and Machine Owners. The Pool Administrator modulate the priority user and job, and the allocation of available resource.

Central Manager, Submit Machine and Execution Machine are there main components in HTCondor [61]. The jobs are submitted from the submit machine, then the Central Manager identifies the feature and usage information of every resource which pool in system. The execution machine is decided by the Central Manager by matchmaking to Job Submitters and Machine Owners.

Unlike other DGC middleware, HTCondor implements the push model, so the central manager can send the jobs to the vacant client even without any request from the client.

### 3.2.4 SZTAKI Local Desktop Grid (LDG)

SZTAKI (The Computer and Automation Research Institute of the Hungarian Academy of Sciences) LDG is start in 2005 where connected widespread any small and medium desktop grid as one is the main objective of this middleware. This idea is result from the common paradigm of desktop grid computing where many resource providers – few users [58], [62], means even there are many volunteers who want to allocate the desktop to be part of desktop grid but in the same time just a few users can utilize its. It is also evident from the reality where sometimes in one institution almost in each department has its own local desktop grid infrastructure, but unfortunately, they don't connect each other [62].

In consequence SZTAKI LDG tries to overcome this problem by introducing the hierarchy way. It means, in the same time each of separate desktop grid can ask or/and send work (push and pull mode) [62]. Looking how it works, this model is almost the same to how the p2p model work where each desktop grid has its own right to send or/and ask the job. The policy of ask and/or send the job is depending the job loading on the desktop grid itself.

SZTAKI LDG uses the derivative of BOINC to handle the desktop grid. Derivative means SZTAKI LDG has make up some new components in BOINC, especially in server side. The feature such as the ability of BOINC to pull jobs from somewhere else in the same hierarchy. Also some modifications in the client side, for instance it has to report the available processor and platform to top of the hierarchy [58]. As security reason, HTTPS protocol is used to communicate and also transfer data that is carried out between client and SZTAKI LDG server [18].

### 3.2.5 QADPZ (Quite Advanced Distributed Parallel Zystem) [63]

QADPZ which is developed using C++ has three main frames to executes its work: slaves, masters, and clients. Slave is the donation resource which is taking part in any computing. Master has task to organize everything that act on job process, for instance start and stop process, queuing the task.

The client is bridging between the user and the system, where the user can create the job, monitoring the status. The job that has been created

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

will be sent to master, then master will distribute it to the slave. QADPZ uses UDP as connection protocol for each that main frames. This UDP Protocol has some advantages [64], such as:

- ✓ The data rate is determined when the application is sent.
- ✓ The application will receive the data as soon as possible even though the data sometimes is not useful.
- ✓ The transport layer will never resend the missing packets.
- ✓ The possibility for the Internet checksum to verify the UDP header and the data payload.

### 3.2.6 Progress Thru Processors

Progress Thru Processor is initiated by Intel in 2009. The aim of this Progress Thru Processor is to embed the BOINC into Facebook application [65], [66]. Intel objectifies this project by collaboration with GridRepublic desktop grid computing. To participate to this Progress Thru Processor, you just need to have Facebook account, and then join the project. While you post the message, read the wall, the Progress Thru Processor shall runs.

## 5. DISCUSSION

The development of desktop grid computing has evolved over 30 years. Some of middlewares have been evolved to improve its function while some of them has already disappeared as shown in Table 2.

*Table 2: Evolution of DGC Middlewares*

| Architecture | Middleware | Date released | Current status |
|---|---|---|---|
| Peer-to-peer | XtremWeb-CH | Unknown | Information not available |
| | OurGrid | 2010 | Latest release 2013 |
| Client server | XtremWeb | 2000 | Latest released 2008. |
| | BOINC | 2002 | Latest released 2020. |
| | HTCONDOR | 1988 | Latest released 2020. |
| | SZTAKI Local Desktop Grid | 2005 | Information not available |
| | QADPZ | 2001 | Latest released 2003 |
| | Progress Thru Processor. | 2009 | Information not available |

Currently the most popular DGC middleware are BOINC and HTCONDOR. BOINC is famous in term of volunteer computing [67] and in other side HTCONDOR is well known as desktop grid middleware. As prominent middleware, BOINC middleware is also available in Google Play Store as Android app where users can easily be a volunteer in any projects as study diseases, predict global warming, or discover pulsars.

The comparison between cloud computing and desktop grid computing is shown in Table 3.

*Table 3: Comparison between Cloud Computing and DGC*

| Issues | Comparisons |
|---|---|
| Utilization Cost | We have to pay for using the cloud computing services whereas DGC is utilizing available resources. So, in this case the use of DGC is cheaper compare to the use of cloud computing. |
| Management Cost | Cloud computing is managed by cloud providers while DGC need to be managed by the organization. So, in term of management the use of cloud computing reduces overhead of an organization. |
| Security | DGC is managed internally, and data will be limited within the organization, and thus will be more secured. |
| Available Resource | The current cloud computing can provide almost unlimited resource, while the available resource for DGC is limited by the number of available desktop computers. |
| Reliability | Both cloud computing and DGC are supported by reliable software technology and thus both of them can be considered to be reliable. |
| Resource Management | DGC is available by the probability of heterogeneity in terms of operating system, bandwidth and availability. But in other side cloud computing is available by predictable operating system, bandwidth and availability. |

From Table 3, it seems that there are some advantages and disadvantages of using DGC as compared to using cloud computing. Since most organizations, especially institutions of higher learning have a lot of desktop computers that are not fully used at all time, DGC can still be utilized to solve problems that requires high computational

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

power. Some examples of current projects that involve the use of DGC are [13].

Another area where DGC can be a good option if the application involves high security data. Since cloud computing is managed by people outside the organization, data security cannot be fully guaranteed. There are many examples of data leakage due to security flaws of cloud computing providers. A 2017 study by CGI and Oxford Economics measured the costs resulting from data breaches in the last five years at more than $50 billion [68]. DGC is managed internally, and data will be limited within the organization, and thus will be more secured.

The third area where DGC can be fully utilized is to support the concept of volunteer computing [16]. Volunteer computing (VC) is a kind of distributed computing that make use of the aggregated spare computing resources from smartphone and laptop [69]. Those spare or idle resources are usually donated by common people for scientific project [70]. The first well-known [12] and most popular VC project is SETI@Home that was launched in 1999 and managed by a group of researchers at the Space Sciences Laboratory of the University of California, Berkeley. The purpose of this project is to use large-scale distributed computing to perform a sensitive search for radio signals from extraterrestrial civilizations [71]. The other projects are GIMPS that was started in 1996. The aim of this project is to find the largest prime number. The project has engaged 182,795 volunteers who provided 1,614,096 CPUs and the largest prime number found so far has 24,862,048 digits. ATLAS@Home is a research project that uses volunteer computing to run simulation of particle physics experiments at CERN. The others latest project is theSkyNet Pan-STARRS1 Optical Galaxy Survey (POGS) project [72].

Another development of DGC is in the use of DGC middlewares on other devices such as smartphone to provide the required computing power. DreamLab [73], for example, is a project utilizing BOINC middleware and runs on iPhone and iPad. Currently DreamLab has more than 100,000 users [74]. The aim of this project is to uses the processing power of idle phones to help solve cancer problem. Another project which utilize volunteered spare resource of smartphone is ALICE Connex [75]. Similar to DreamLab, ALICE Connex also stand behind BOINC middleware.

## 6. DIFFERENT FROM PRIOR WORK

The purpose of this work is to give insight regarding how DGC can be an option other than CC. The option here does not mean that DGC can replace for CC, but in some aspect and circumstance DGC is better option compare to CC, for example in data security. This work tries to give insights to make user of DGC or CC can make a choice whether to use DGC or CC that depend on their circumstance or issue.

## 7. CONCLUSION

DGC as a part of Grid Computing is worth to be consider as an option beside CC. Even it is unequal head to head comparison, but in some terms, DGC is outperforming the CC. In term of security issue DGC can be surely better than CC. This alone can show how the future direction of DGC where nowadays security is a big challenge and a reason in considering to adopting CC. There was a lot of DGC middleware but just a few is still alive and in service.

The use of desktop grid computing technology can still be utilized for solving computing intensive application. With the growth of desktop computer belonging to an organization and the increasing of internet penetration worldwide, desktop grid computing can be a good alternative to provide the cheapest high-power computing. DGC, which is a simple yet effective computing system can also be a good option to be utilized if an application involved some sensitive and secured data.

Another major use of DGC is to support volunteer computing projects. Some of volunteer computing projects currently available are SETI@Home, GIMPS and Atlas@Home. The use of volunteer computing is expected to be more popular in the future since devices that can be volunteered is not only limited to desktop computers but also other computing devices such as smart phones. The idea of volunteering computing devices while we are asleep in order to solve problems that can give benefit to humanity, such as DreamLab, is interesting and will certainly get support from a lot of volunteers.

## REFERENCES:

[1] A. Kaminsky, *Big CPU, Big Data: Solving the World's Toughest Computational Problems with Parallel Computing*, 1st ed. USA: CreateSpace Independent Publishing Platform, 2016.

[2] M. U. Bokhari, Q. Makki, and Y. K. Tamandani, "A Survey on Cloud Computing," in *Big Data Analytics*, 2018, pp. 149–164.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[3] A. L. S. Saabith, E. A. Sundararajan, and A. A. Bakar, "Parallel implementation of Apriori algorithms on the Hadoop-MapReduce platform - An evaluation of literature," *J. Theor. Appl. Inf. Technol.*, vol. 85, pp. 321–351, 2016.

[4] W. Tian and Y. Zhao, "1 - An Introduction to Cloud Computing," in *Optimized Cloud Resource Management and Scheduling*, W. Tian and Y. Zhao, Eds. Boston: Morgan Kaufmann, 2015, pp. 1–15.

[5] D. P. Anderson, "Volunteer computing: The ultimate cloud.," *ACM Crossroads*, vol. 16, no. 3, pp. 7–10, 2010.

[6] B. Krašovec and A. Filipčič, "Enhancing the Grid with Cloud Computing," *J. Grid Comput.*, vol. 17, no. 1, pp. 119–135, Jan. 2019.

[7] M. Kayali, N. Mohd Satar, and M. Mukhtar, "Adoption of Cloud Based E-learning in Lebanon: Examining the Mediating Role of Attitude," 2020.

[8] M. H. Kayali, N. Safie, and M. Mukhtar, "Adoption of cloud based E-learning: a systematic literature review of adoption factors and theories," *J. Eng. Appl. Sci.*, vol. 11, no. 8, pp. 1839–1845, 2016.

[9] L. Bölöni and D. Turgut, "Value of information based scheduling of cloud computing resources," *Futur. Gener. Comput. Syst.*, vol. 71, pp. 212–220, Jun. 2017.

[10] A. Shakeabubakor, E. Sundararajan, and A. Razak Hamdan, "Cloud Computing Services and Applications to Improve Productivity of University Researchers," *Int. J. Inf. Electron. Eng.*, vol. 5, no. 2, 2015.

[11] S. R. Chakravarthy and A. Rumyantsev, "Efficient Redundancy Techniques in Cloud and Desktop Grid Systems using {MAP}/G/c-type Queues," *Open Eng.*, vol. 8, no. 1, pp. 17–31, Mar. 2018.

[12] I. Chernov, N. Nikitina, and E. Ivashko, "Task Scheduling in Desktop Grids: Open Problems," *Open Eng.*, vol. 7, no. 1, pp. 343–351, 2017.

[13] E. Morozov, O. Lukashenko, A. Rumyantsev, and E. Ivashko, "A Gaussian approximation of runtime estimation in a desktop grid project," in *2017 9th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, 2017, pp. 107–111.

[14] P. K. D. Pramanik, P. Choudhury, and A. Saha, "Economical supercomputing thru smartphone crowd computing: An assessment of opportunities, benefits, deterrents, and applications from India's perspective," in *2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS)*, 2017, pp. 1–7.

[15] S. Choi *et al.*, "Characterizing and Classifying Desktop Grid," in *Seventh IEEE International Symposium on Cluster Computing and the Grid (CCGrid '07)*, 2007, pp. 743–748.

[16] Y. Kolokoltsev, E. Ivashko, and C. Gershenson, "Improving 'tail' computations in a boinc-based desktop grid," *Open Eng.*, vol. 7, no. 1, pp. 371–378, 2017.

[17] O. Nov, D. Anderson, and O. Arazy, "Volunteer Computing: A Model of the Factors Determining Contribution to Community-based Scientific Research," in *Proceedings of the 19th International Conference on World Wide Web*, 2010, pp. 741–750.

[18] A. C. Marosi, Z. Balaton, P. Kacsuk, and D. Drótos, "SZTAKI Desktop Grid: Adapting Clusters for Desktop Grids," in *Remote Instrumentation and Virtual Laboratories*, 2010, pp. 133–144.

[19] C. Yang, Q. Huang, Z. Li, K. Liu, and F. Hu, "Big Data and cloud computing: innovation opportunities and challenges," *Int. J. Digit. Earth*, vol. 10, no. 1, pp. 13–53, 2017.

[20] J. W. Rittinghouse and J. F. Ransome, *Cloud Computing: Implementation, Management, and Security*. CRC Press, 2016.

[21] N. H. binti Azizul, A. bin Mohd Zin, and E. Sundararajan, "The design and implementation of middleware for application development within honeybee computing environment," *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 6, no. 6, pp. 9137–9945, 2016.

[22] K. Geihs, "Middleware challenges ahead," *Computer (Long. Beach. Calif).*, vol. 34, no. 6, pp. 24–31, Jun. 2001.

[23] S. M. Sadjadi and P. K. McKinley, "A survey of adaptive middleware," *Michigan State Univ. Rep. MSU-CSE-03-35*, p. 11, 2003.

[24] R. E. Schantz and D. C. Schmidt, "Research Advances in Middleware for Distributed Systems: State of the Art," in *Communication Systems: The State of the Art IFIP 17th World Computer Congress --- TC6 Stream on Communication Systems: The State of the Art August 25--30, 2002, Montréal, Québec, Canada*, L. Chapin, Ed. Boston, MA: Springer US, 2002, pp. 1–36.

[25] T. A. Bishop and R. K. Karne, "A Survey of Middleware.," in *Computers and Their Applications*, 2003, pp. 254–258.

[26] M. Khan, T. Mahmood, and S. Hyder, "Scheduling in desktop grid systems: Theoretical evaluation of policies and frameworks," *Int. J. Adv. Comput. Sci. Appl.*, vol. 8, no. 1, pp. 119–127, 2017.

[27] G. C. Silaghi, F. Araujo, L. M. Silva, P. Domingues, and A. E. Arenas, "Defeating Colluding Nodes in Desktop Grid Computing

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Platforms," *J. Grid Comput.*, vol. 7, no. 4, p. 555, Aug. 2009.

[28] F. Araujo, P. Domingues, D. Kondo, and L. M. Silva, "Validating Desktop Grid Results By Comparing Intermediate Checkpoints," in *Achievements in European Research on Grid Systems: CoreGRID Integration Workshop 2006 (Selected Papers)*, S. Gorlatch, M. Bubak, and T. Priol, Eds. Boston, MA: Springer US, 2008, pp. 13–24.

[29] K. Watanabe, M. Fukushi, N. Funabiki, and T. Nakanishi, "Performance Evaluation of Check-By-Voting for Colluding Attack in Volunteer Computing Systems," in *IAENG Transactions on Engineering Technologies: Special Issue of the International MultiConference of Engineers and Computer Scientists 2012*, G.-C. Yang, S.-I. Ao, X. Huang, and O. Castillo, Eds. Dordrecht: Springer Netherlands, 2013, pp. 33–48.

[30] K. Watanabe, M. Fukushi, and S. Horiguchi, "Optimal Spot-checking for Computation Time Minimization in Volunteer Computing," *J. Grid Comput.*, vol. 7, no. 4, p. 575, Aug. 2009.

[31] M. P. M. D. Sharma and M. P. Mittal, "Job scheduling algorithm for computational grid in grid computing environment," *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 3, no. 5, 2013.

[32] L. Ni and A. Harwood, "An Adaptive Checkpointing Scheme for Peer-to-Peer Based Volunteer Computing Work Flows," in *2008 Ninth International Conference on Parallel and Distributed Computing, Applications and Technologies*, 2008, pp. 227–234.

[33] N. Andrade, L. Costa, G. Germóglio, and W. Cirne, "Peer-to-peer grid computing with the ourgrid community," in *Proceedings of the SBRC*, 2005, pp. 1–8.

[34] C. Briquet and P.-A. de Marneffe, "Learning Reliability Models of Grid Resource Supplying," *CGW'05 Proc.*, 2005.

[35] E. de Lucena Falcão, F. Brasileiro, A. Brito, and J. L. Vivas, "Enhancing fairness in P2P cloud federations," *Comput. Electr. Eng.*, vol. 56, pp. 884–897, 2016.

[36] I. Wu *et al.*, "A Volunteer-Computing-Based Grid Environment for Connect6 Applications," in *2009 International Conference on Computational Science and Engineering*, 2009, vol. 1, pp. 110–117.

[37] G. Fedak *et al.*, "EDGeS: A Bridge between Desktop Grids and Service Grids," in *The Third ChinaGrid Annual Conference (chinagrid 2008)*, 2008, pp. 3–9.

[38] Y.-K. R. Kwok, *Peer-to-peer computing: applications, architecture, protocols, and challenges*. Crc Press, 2011.

[39] N. Jacq, *From Genes to Personalized Healthcare: Grid Solutions for the Life Sciences : Proceedings of HealthGrid 2007*. IOS Press, 2007.

[40] C. Cérin and G. Fedak, *Desktop grid computing*. Chapman and Hall/CRC, 2012.

[41] B. B. Mohamed, N. Marko, and A. Nabil, "Programming distributed medical applications with XWCH2," *Stud. Health Technol. Inform.*, vol. 159, no. Healthgrid Applications and Core Technologies, pp. 100–111, 2010.

[42] N. Abdennadher and R. Boesch, "Towards a Peer-To-Peer Platform for High Performance Computing," in *Advances in Grid and Pervasive Computing*, 2007, pp. 412–423.

[43] N. Abdennadher, M. Niinimaki, and M. BenBelgacem, "The XtremWebCH Volunteer Computing Platform," in *Desktop Grid Computing*, Chapman and Hall/CRC, 2012, pp. 79–104.

[44] W. Cirne *et al.*, "Running Bag-of-Tasks applications on computational grids: the MyGrid approach," in *2003 International Conference on Parallel Processing, 2003. Proceedings.*, 2003, pp. 407–416.

[45] W. Cirne *et al.*, "Labs of the World, Unite!!!," *J. Grid Comput.*, vol. 4, no. 3, pp. 225–246, Sep. 2006.

[46] F. Brasileiro, E. Araujo, W. Voorsluys, M. Oliveira, and F. Figueiredo, "Bridging the High Performance Computing Gap: the OurGrid Experience," in *Seventh IEEE International Symposium on Cluster Computing and the Grid (CCGrid '07)*, 2007, pp. 817–822.

[47] G. F. Coulouris, J. Dollimore, and T. Kindberg, *Distributed systems: concepts and design*. pearson education, 2005.

[48] G. Fedak, C. Germain, V. Neri, and F. Cappello, "XtremWeb: a generic global computing system," in *Proceedings First IEEE/ACM International Symposium on Cluster Computing and the Grid*, 2001, pp. 582–587.

[49] A. Luntovskyy and J. Spillner, "Architectural Transformations in Distributed Systems," in *Architectural Transformations in Network Services and Distributed Systems*, Wiesbaden: Springer Fachmedien Wiesbaden, 2017, pp. 13–44.

[50] N. Scientist, *The End of Money: The story of bitcoin, cryptocurrencies and the blockchain revolution*. John Murray Press, 2017.

[51] M. Swan, *Blockchain: Blueprint for a New Economy*. O'Reilly Media, 2015.

[52] A. Holohan, *Community, Competition and Citizen Science: Voluntary Distributed Computing in a Globalized World*. Taylor & Francis, 2016.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[53] Cameron, David, Field, Laurence, Giannakis, Nikolas, and H\oimyr, Nils, "Extending CERN computing to volunteers - LHC@home consolidation and outlook," *EPJ Web Conf.*, vol. 214, p. 3016, 2019.

[54] T. M. Mengistu and D. Che, "Survey and taxonomy of volunteer computing," *ACM J. Comput. Surv*, pp. 1–35, 2019.

[55] J. Rantala and R. Piché, "Software Systems for Distributed Scientific Computing," 2009.

[56] D. P. Anderson, E. Korpela, and R. Walton, "High-performance task distribution for volunteer computing," in *First International Conference on e-Science and Grid Computing (e-Science'05)*, 2005, p. 8 pp.-pp.203.

[57] J. D. Baldassari, "Design and evaluation of a public resource computing framework," 2006.

[58] Z. Balaton *et al.*, "SZTAKI Desktop Grid: a Modular and Scalable Way of Building Large Computing Grids," in *2007 IEEE International Parallel and Distributed Processing Symposium*, 2007, pp. 1–8.

[59] E. Imamagic, B. Radic, and D. Dobrenic, "An approach to grid scheduling by using condor-G matchmaking mechanism," *J. Comput. Inf. Technol.*, vol. 14, no. 4, pp. 329–336, 2006.

[60] M. J. Litzkow, M. Livny, and M. W. Mutka, "Condor-a hunter of idle workstations," 1987.

[61] C. Chapman *et al.*, "Condor services for the Global Grid: Interoperability between Condor and OGSA," UK Engineering and Physical Science Research Council, 2004.

[62] P. Kacsuk *et al.*, "Sztaki desktop grid: A hierarchical desktop grid system," in *Cracow'06 Grid Workshop*, 2006.

[63] M. Vl\uadoiu and Z. Constantinescu, "Development journey of QADPZ-A desktop grid computing platform," *Int. J. Comput. Commun. Control*, vol. 4, no. 1, pp. 82–91, 2009.

[64] L.-A. Larzon, M. Degermark, and S. Pink, *UDP lite for real time multimedia applications*. Hewlett-Packard Laboratories, 1999.

[65] V. Chang, "A cybernetics Social Cloud," *J. Syst. Softw.*, vol. 124, pp. 195–211, 2017.

[66] A. McMahon and V. Milenkovic, "Social volunteer computing," *J. Syst. Cybern. Informatics*, vol. 9, no. 4, pp. 34–38, 2011.

[67] M. Posypkin and N. Khrapov, "Branch and bound method on desktop grid systems," in *2017 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, 2017, pp. 526–528.

[68] Reuters, "Cyber Breaches Cause Permanent Damage to Stock Market Value," *Fortune*. Fortune, Apr-2017.

[69] E. Lavoie and L. Hendren, "Personal Volunteer Computing," in *Proceedings of the 16th ACM International Conference on Computing Frontiers*, 2019, pp. 240–246.

[70] S. Alonso-Monsalve, F. García-Carballeira, and A. Calderón, "ComBos: A complete simulator of Volunteer Computing and Desktop Grids," *Simul. Model. Pract. Theory*, vol. 77, pp. 197–211, 2017.

[71] E. Korpela, V. Gajjar, J. Cobb, D. Anderson, and D. Werthimer, "SETI@ home analysis of observations of 220 nearby stars," in *42nd COSPAR Scientific Assembly*, 2018, vol. 42.

[72] K. Vinsen and D. Thilker, "A BOINC11Berkeley Open Infrastructure for Network Computing. based, citizen-science project for pixel spectral energy distribution fitting of resolved galaxies in multi-wavelength surveys," *Astron. Comput.*, vol. 3–4, pp. 1–12, 2013.

[73] "DreamLab: App creates 'smartphone supercomputer' to help find cure for cancer - ABC News." Nov-2015.

[74] "Corona-AI Project Asks DreamLab App Users to Help Create 'Virtual Supercomputer' to Assist in COVID-19 Research Efforts." Apr-2020.

[75] P. Jenviriyakul, G. Chalumporn, T. Achalakul, F. Costa, and K. Akkarajitsakul, "ALICE Connex: A volunteer computing platform for ALICE experiments," in *2016 International Conference on Big Data and Smart Computing (BigComp)*, 2016, pp. 300–303.