Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2013) 21: 1646 – 1664
© TÜBİTAK
doi:10.3906/elk-1108-27

Four-dimensional model for describing the status of peers in peer-to-peer distributed systems

Seyedeh Leili MIRTAHERI,[1,*] Ehsan Mousavi KHANEGHAH,[1] Mohsen SHARIFI,[1] Behrouz MINAEI-BIDGOLI,[1] Bijan RAAHEMI,[2] Mohammad Norouzi ARAB,[1] Abbas Saleh ARDESTANI[3]
[1]School of Computer Engineering, Iran University of Science and Technology, Tehran, Iran
[2]School of Information Technology and Engineering, University of Ottawa, Ottawa, Ontario, Canada
[3]Department of Business Management, Faculty of Management, Islamic Azad University Central Tehran Branch, Tehran, Iran

Received: 09.08.2011 • **Accepted:** 15.03.2012 • **Published Online:** 02.10.2013 • **Printed:** 28.10.2013

Abstract: One of the important aspects of decision making and management in distributed systems is collecting accurate information about the available resources of the peers. The previously proposed approaches for collecting such information completely depend on the system's architecture. In the server-oriented architecture, servers assume the main role of collecting comprehensive information from the peers and the system. Next, based on the information about the features of the basic activities and the system, an exact description of the peers' status is produced. Accurate decisions are then made using this description. However, the amount of information gathered in this architecture is too large, and it requires massive processing. On the other hand, updating the information takes time, causing delays and undermining the validity of the information. In addition, due to the limitations imposed by the servers, such architecture is not scalable and dynamic enough. The peer-to-peer architecture was introduced to address these concerns. However, due to a lack of complete knowledge of the peers and the system, the decisions are made without a precise description of the peers' status and are only based on the hardware data collected from the peers. Such an abstract and general image of the peers is not adequate for the purpose of decision making. In this paper, a 4-dimensional model is presented for the purpose of information collection and the exact description of the peer's status, including the features of the peer, the basic activity, the time, and the specifications of the system. The proposed model is for a server-oriented architecture, but it also adapts to the peer-to-peer serverless architecture. Based on this model, a new approach is introduced for information collection and an exact description of the peers' status in a peer-to-peer system based on the Latin square concept. We evaluate the model in the server-oriented and serverless situations. The workload is considered as the basic activity in our evaluation. Our evaluation demonstrates that in a server-oriented situation, increasing the size of the system has a direct relation with time. However, a serverless situation does not follow this behavior.

Key words: Distributed systems, peer-to-peer systems, server-oriented, workload, Latin square, serverless

1. Introduction

The most important concern about the decisions made in distributed systems for resource management, load balancing, task distribution, and so on is collecting accurate and updated data about the resources of the peers [1–3]. In the real world, the decision-making issue calls for 3 categories of information: 1) information that the decision maker should gather regarding the system's elements, 2) information about the activity for which the

*Correspondence: mirtaheri@iust.ac.ir

decision is to be made, and 3) information about the system's status or environment of the decision maker. These 3 groups of data should also be gathered for the decisions made in distributed systems. The decision maker should collect information about the system elements, peers, and the features of the activity that should be done in this system (we name it "basic activity"). In addition, data about the structure and the basic features of the system need to be collected. Based on these data, the decision maker then generates descriptions of the peers to be used in making its decisions.

Collecting information about the available resources of the peers in a distributed system should be done according to the system's goal, which directs the task of decision making. Based on the model proposed by Foster about the features of the resources, the range of resources that can be shared in a distributed system spans a wide range, from those with thoroughly dynamic features to those with completely static features [4]. Information collected about the resources available in a system depends on the system's goals. For example, in systems with computational sharing goals, the information related to the CPU load and memory status are collected [5,6]. The systems whose goal is data sharing collect data about databases and XML files [7,8]. Systems designed and implemented with the goal of file sharing collect data pertinent to files. Such systems may limit their search and administration to a few specific types of files [9,10]. Systems designed to perform input/output (I/O) sharing and collaboration among the nodes collect information about the desired I/O [11,12]. As such, in any arbitrary distributed system with any arbitrary architecture, the crucial information is gathered in accordance with their major goal. Such types of information have specific features: their volume is large, they respond to only a few specific requests, and they are merely collected by the hardware approach. Since the data are collected based on a series of parameters and certain features, making decisions calls for the relations between these parameters and features and the effects they exert upon each other. Thus, there must be a model by which data are collected and decisions are made.

Collecting information in distributed systems largely depends on the system's architecture [13]. The existing architectures may be divided into 2 major categories: server-oriented and peer-to-peer [14]. In the server-oriented architecture, a monitoring model is used for the purpose of collecting data and making decisions [15]. The server is aware of the status of each of the members within the system [16]. However, collecting information can be done by 2 methods: 1) members send the data indicating their own status at specific time intervals or upon a specific event and 2) the server takes action and gathers the required information indicating the member's status at specific time intervals or upon a specific event [17,18]. In both methods, the server is constantly adequately informed about the members, the encompassing system, and the activity it desires to make decisions about. However, information collection in the server-oriented architecture suffers from 3 problems, despite its several advantages: scalability, dynamism, and the huge amount of information collected [16].

Peer-to-peer with serverless architecture was introduced to overcome these problems. All of the peers have the same role. There is no dependency between the peers, or such dependency is maintained at the lowest degree possible. This property makes the system more scalable and dynamic [19,20]. However, the major problem here is the lack of accurate data covering the system and the status of each member. There are 2 general methods for data collection in this architecture [21]: 1) case-wise, requests are either randomly sent to other peers or a heuristic is used to choose the target peer (e.g., learning-oriented, best neighbor, a combination of the 2, etc.) [22]; and 2) periodic or event-based: resource information is sent periodically or upon the occurrence of a specific event [23,24]. Of course, this information merely describes a peer regarding its hardware resources. Consequently, reliability and response time are 2 major challenges of this architecture [25].

In fact, the implementations only consider the peers' status and describe them from a hardware standpoint. Regardless of the basic activity, responding to requests as time goes by and only relying on the local manner and information, each peer performs information collection. Therefore, the data gathered here offer an abstract definition of the status of each peer [26].

If the 4-dimensional model is used in a distributed manner across the peer-to-peer architecture, with smaller amounts of data collected and higher accuracy gained, the peers' status can be described. Based on the 4-dimensional model, each peer describes its own status. To describe the status, the first thing each peer needs to investigate is its own hardware resources. Such parameters or features are referred to as machine attributes (MAtrib) in this paper. In addition, each peer needs to consider the system within which it resides. System attributes (SAtrib) refer to these system parameters, constituting another dimension of the model. Furthermore, each peer must describe its status regarding the basic activity through which it is involved. This assumption is necessary since further decisions are made according to a basic activity already defined in the system. Activity attributes (AAtrib) include all such features relating to the basic activity. Finally, the time dimension has been incorporated into the model to represent the changes occurring to each peer's status over time.

The 4-dimension model relies on a good understanding of the relations between the dimensions. Therefore, we need to use 3-dimensional data structures for saving the information in them, such that we can offer a precise image of the peer's status. On the other hand, the generation and maintenance of the information in 3-dimensional data structures is too difficult and costly. Moreover, the storage and retrieval is complicated. In this paper, a solution has been presented, which is to use a Latin square combination design. Thus, the matrix describing the peer's status, entitled 'Peer's Square', is used throughout this paper. The 4-dimensional model is put into action by launching it on the unstructured peer-to-peer system, with the basic activity defined as the workload.

The 4-dimensional model offered for describing the peer's status has the following advantages and merits:

1. The presented model takes into account the parameters SAtrib and AAtrib, including the basic activity and the system's most effective parameters. These considerations, along with the incorporated time dimension, provide a more accurate description of peers. This property significantly reduces the error while making decisions, a situation that frequently occurs in peer-to-peer systems.

2. This model does not require a great deal of data to be gathered for defining the peer's status, the opposite of the situation that occurs in server-oriented systems.

3. Due to the small amount of data collected by this distributed 4-dimensional model, less time is required to process it. In addition, less time is required to describe the peer's status.

4. Since peers have the same role, there exists no communication overhead between the server and the member peers. Therefore, the peers' status will be described with higher precision in comparison to the server-oriented systems.

The rest of the paper is organized as follows: in Section 2, the principle and definition concepts are presented. The Latin square combination design is discussed in Section 3. The structural features of the peer-to-peer system, the workload activity, the 4-dimensional model, and the associated matrix are discussed in detail in Section 4. The implementation of the Latin square is presented in Section 5. Section 6 provides the experimental results. Finally, we conclude the paper in Section 7.

2. Primary concepts

In this section, the primary concepts and the definition of the model and the Latin square are presented.

2.1. Four-dimensional model

In this section, we describe the model's methods of information gathering and decision making in peer-to-peer systems relying on a server-oriented architecture. By examining the current decision-making models incorporated into the server-oriented architecture, it is revealed that servers make decisions by creating special combinational designs [27]. Special combinational designs refer to the combination of 3 feature types: 1) some measurable features whose resultant describes each peer in terms of its hardware characteristics, 2) some features specifying the system, and 3) some features specifying the basic activity that has the most significant effect on the system. Using different combinations of these features, servers make decisions about any event or request.

Since the data are updated through the passage of time, a fourth time dimension is added to the model. Thus, a 4-dimensional model can be represented for data collection in this architecture. Using this architecture incorporating the proposed model, decisions are made reliably to a high extent and the number of unresponded requests is kept at the minimum level. In addition, the response time is optimized.

As can be seen in Figure 1a, in server-oriented systems, the server collects information from peers and obtains an accurate description about the status of each peer based on the nature of the nodes basic activity. In this architecture, the server executes and controls the job based on the collected information.

In a serverless peer-to-peer system, shown in Figure 1b, peers that start a global activity in collaboration with other nodes should jointly collect information about their status, nature of the activity, and the overall system. However, in reality, each peer only collects information of the peers that are working with them on executing an activity.

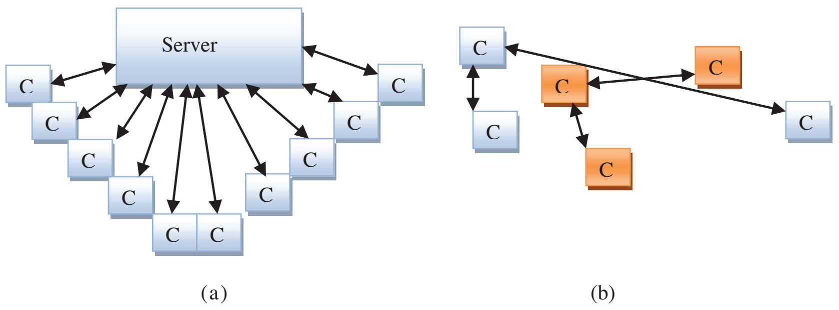

Figure 1. a) Collecting information in server-oriented systems and b) collecting information in peer-to-peer systems.

Definition 1. Describing peer status is defined as obtaining the peer's general and logical scheme regarding a special basic activity, where decision making relies on when the peer is a member of a distributed system. In other words, if we manage to take into account all of the possible states that any peer may attain while conducting an activity (or a set of activities), then examining the peer's status actually means that we are trying to find out a mathematical relation between time passage through the peer's standpoint and the current peer's status. Such states are described regarding the basic activities.

Describing the peer's status includes the following general concepts and definitions, constituting the 4 dimensions of the model:

Definition 2. AAtrib comprises the features of the basic activity that a peer's status is described by and decisions are made based on them.

Definition 3. MAtrib comprises the features of the peer in terms of any relevant hardware and software involved.

Definition 4. SAtrib comprises effects and draws results from the system encompassing the peer. Such effects may not be neglected.

Definition 5. Time, as the system is constantly changing and thus examining such changes through time passage, is important.

2.2. Latin square combination design

As already mentioned, we need a mathematical relation between these 4 spaces; therefore, the combinational design is arranged to establish these relations. If we are to implement such relations or states, the 3-dimensional data structures have to be used to get a precise description of the peer's status. Using data structures with fewer dimensions (e.g., 2 dimensions) will result in losing one or more relations between these spaces. Data generation and maintenance inside 3-dimensional data structures is difficult and costly and presents complexity in their retrieval. Applying the Latin square combination design [28] is the solution proposed for overcoming this problem.

The Latin square helps us to consider several relations of inherent codependency without repeated states for the processes. In this paper, we have used 2 basic features of the Latin square: the lack of any repeated a_{ij} in the rows or columns and the capability for examining various states of any combination design within a 2-dimensional space. The fact that there is no repeated a_{ij} in the rows or columns has the advantage that the square evades all of the repeated states when it describes the peer's status.

The Latin square is a finite combinational design. It means that this square, with all of the information regarding the peer's status (MAtrib), features of the basic activity (AAtrib), time (either explicitly or implicitly), and the system's factors (SAtrib), is capable of defining the state that the peer enters upon occurrence of a special event. In other words, the Latin square is a structure that provides the opportunity for describing a peer's status based on the peer's resources, a certain activity, and the encompassing system through time.

Practically, the effect(s) of AAtrib on any of the other 3, i.e. time, SAtrib, and MAtrib, may be identified pairwise. However, such identification investigates the effects of AAtrib on the other 3 in a separate manner, resulting in a definition in which some certain relations are evaded. This means that in the case that a process, using a traditional design, attempts to examine the effect(s) of AAtrib on the peer's status in the form of (MAtrib, SAtrib ,time), the outcome will be 3 separate examinations and 3 different overviews, while with the Latin square represented in this paper, there will be a single examination and only a single overview. The Latin square provides the process with the opportunity of describing the peer's status based on the effects of a specific activity exerted on the 3 sets all together.

3. The 4-dimensional model in the peer-to-peer system based on the workload basic activity

In this section, the suggested model is examined and applied in the distributed peer-to-peer system [29] based on workload as a basic activity. For the first step, we describe the peer-to-peer system characteristics and then we adopt the model to the specific characteristics of the peer-to-peer workload-based system.

The selected peer-to-peer system should be able to execute with an acceptable accuracy in both the server-oriented and serverless peer-to-peer architecture [29]. This property allows us to evaluate the model in different architectures and compare the obtained Latin square. The workload, the selected basic activity, is utilized in load balancing based on the processer and memory resources, and so we need a peer-to-peer system with resource sharing capabilities. An important property of the peer-to-peer system is its implementation level. Since the peer-to-peer system is implemented in the kernel of an operation system, the data structure can be easily accessed, which in turn facilitates accurate information retrieval for MAtrib. To apply this model, we also need to have complete information about the structure and functionality of the selected peer-to-peer system to form SAtrib. The peer-to-peer system's structure is based on a mathematical model that will accommodate the application of the 4-dimensional model.

3.1. The distributed peer-to-peer system

The distributed peer-to-peer system [29] has no primary structure and the execution task definition (basic execution task, known as the basic activity of the system) is always considered a higher priority compared to the system's structure definition. Actually, the system's structure assumes its shape relying on the work performed (request made). This system has 3 eras: Mosaic, Jurassic, and Kertaseh. The system's structure is formed during these 3 eras.

There are 4 types of resources defined in the incorporated model of this specific system, as there are in common operating systems, which are I/O, Memory, File, and Process. The defined regions in the system are formed based on the definitions provided to these 4 types of resources.

In the peer-to-peer system, each peer has a unit named Oasis, which plays a very crucial role by maintaining the history of the resources' accesses, and based on the classification pattern already defined in the system, the resources' locations are identified. Part of each peer's memory space is used to maintain histories relating to these 4 groups of resources and, correspondingly, there are 4 Oasis subadministrative units named File Management Oasis, Memory Management Oasis, I/O Management Oasis, and IPC Management Oasis.

Oasis space contains metadata about peer-to-peer system resources, which are controlled by a peer-to-peer system administrator located on each peer. The access history, resource requests and responses, and resource response routines are kept there. In other words, when a request cannot be satisfied locally for each resource type, a special Oasis space is formed.

The system is based on a supply/demand system. A supply/demand system continuously changes over time; sometimes it is supply-oriented, i.e. the supply is greater than the demand, and sometimes it is demand-oriented. Therefore, the peer-to-peer system structure changes accordingly, which results in different variations of the management rules and logical schemas.

During its formation within the Mosaic era, the system is like a network system. Peers enter the Jurassic era under 2 conditions: 1) upon the occurrence of PBang X, PBang X indicates the condition when a request cannot be satisfied locally; or 2) when a member is forwarded a request by a peer in which PBang X has occurred. Under such conditions, a unit is activated to satisfy the request and logical regions are formed across the system. The region is a logical concept that is formed upon a request and, depending on the resource type requested, a peer is chosen as the coordinator, which will facilitate the implementation of the system management rules across the region.

Within the Kertaseh era, the system is of relative equilibrium and its structure is of stability. This will be true until a certain event happens, after which the system temporarily outbalances and its structure becomes unstable. As soon as supplies and demands are balanced, the system enters a new equilibrium condition and its structure undergoes some changes. In the new condition, the system's management rules correspondingly change. Since in this paper the management rules within the Jurassic era are of interest, some more in-detail descriptions of this era will be presented.

Therefore, the system has 3 different management layers that are defined based on the era within which it resides. Since one of the most basic rules for designing the system is to preserve the autonomy of the peers, all of the management rules will be defined taking into account this principle. In this paper, the method introduced to preserve the peers' autonomy is that each peer describes its own status such that this main principle is not violated.

3.2. Factors affecting the peer's status regarding workload activity in the peer-to-peer system

In this section, we are going to examine the peer's definition using the 4-dimensional model introduced and the basis activity defined as the workload. Since one of the system goals is to avoid highly loaded peers, each peer can be reasonably loaded by providing an exact description of its status. If in describing a peer's status, the number of jobs assigned to it, the number of jobs it successfully completed, or the density of the current jobs and so on are merely considered, then it is like describing the peer's status in an abstract space. In other words, the surrounding system and the peer's role are both neglected. When applying the model presented, we first need to specify the MAtrib, AAtrib, and (MAtrib, AAtrib) sets, which further need describing of the peer's status regarding the basic activity. At the second stage, we describe the system's structural description, and, finally, we need to investigate the basic activity effect on the peer's status through the passage of time. Having all of these, the describing matrix will be formed and one can claim that the peer's description is completely based on the peer's nature and its role in the system.

As mentioned earlier, in order to implement the described matrix, we use a special combinational design named the extended Latin square. The Latin square enables the processes of the local peer to describe the peer's status in accordance with the overall system. It is crucial to notice that the Latin square version used in this research is an extended one, derived from the Latin square known in mathematical literature, and it is adapted to the distributed peer-to-peer system. This version is called the LTE.

3.3. MAtrib set in the peer-to-peer system

The MAtrib set is formed using the same definitions represented for the resources and its classification in any distributed system. As mentioned before, in the applied model, there are 4 types of resources: I/O, Memory, File, and Process. Based on the resource types, various regions are formed across the system. To describe its status, each peer constructs 4 combinational structures corresponding to each resource type based on the workload activity,

From an operating system standpoint, there must be a set of system calls through which users can make use of the 4 types of resources across the system. Moreover, there must be the possibility for a migrated global process (or some of its subprocesses) to satisfy its (their) resource requirements through the system calls provided in the target peer.

The basic operating systems used in the system are those of the UNIX family. Therefore, system calls used in this family are considered as basic system calls. Thus, 4 sets, MAtrib_{IO}, MAtrib_{F}, MAtrib_{M}, and MAtrib_{P}, can be defined and they contain common system calls existing in the UNIX family of operating systems. A process needs the peer's status to make decisions about the workload. In such a condition, the peer' status can be described using 1 of the 4 sets below. This set is determined by the general event (initiation of a migrated global process) that happens in the peer.

3.4. AAtrib set in the peer-to-peer system

Any global process is 1 of the 4 types of processes already defined in the system. Therefore, the inability of the local operating system in satisfying the resource requirements of a process causes a global process to be initiated. Based on the type of the resources required, the type of the global process is determined. On the other hand, the operating system kernel structures provide a process with the ability to make decisions about the local AAtrib set containing the basic features of the basic activity. In this paper, as mentioned, the workload is regarded as the basic activity. Thus, the major goal of this paper is to describe the peer's status based on its workload. Therefore, any decision made by the local processes is towards maintaining the workload. They perform such maintenance by deciding whether the peer can participate in a certain global operation or not.

It should be noted that not only must the AAtrib set represent features of the basic activity, but it must also consider some dependencies that exist between the basic activity and the underlying system. Hence, some features of the system can be seen in this set, as well.

Definition 6. Workload means the number of processes being executed in the peer. Since peers are involved in a distributed system, 2 types of workloads can be considered for each peer, the first regarding local processes (local processes' workload) and the other regarding global processes being executed on the peer (global processes' workload). However, such structures lack the ability to support decisions about the global processes' workload.

To clarify to the members of the AAtrib set in regard to the definition brought for the MAtrib, a definition of the workload activity must be represented. Eq. (1) represents the new definition for the workload activity within any instance of the peer's lifetime.

$$\text{Workload}\,(t) = \frac{\text{Number of System}\,Call_X(t)}{\text{Total Number of System Calls}\,(t)} \qquad \text{For Global Processes} \tag{1}$$

Based on Eq. (1), each member of the AAtrib set is calculated. Two points should be pointed out about Eq. (1): 1) The AAtrib set is a time series. In other words, time is implicitly associated with each member of the set. Parameter 't' in Eq. (1) refers to this property. 2) The AAtrib members should also represent basic features of the surrounding system. Actually, based on Eq. (1), 4 sets are generated: AAtrib_{IO}, AAtrib_F, AAtrib_M, and AAtrib_P, and there is a one-to-one relation between these and those of MAtrib. These relations indicate that each peer is merely described regarding the basic activity involved.

It is worth mentioning that any $(\text{MAtrib}_X, \text{AAtrib}_X)$ pair indicates one element in the describing matrix. When describing the Latin square, we will see that this pair actually represents one single cell in the Latin square, marked as X.

3.5. SAtrib set in the peer-to-peer system

Each peer must reside within a system so that its status can be described. Concerning the peer's functionality and users querying the peer, certainly a system may be considered as allocated to the peer and, consequently, an environment will be dedicated to it, as well. This means that in order to describe a peer's status, the system parameters affecting the peer status must be taken into account. However, such parameters originate from the system's basic features.

In studying the important and basic features of the system, it should be noted that first, the system is a distributed peer-to-peer system with no predescribed structures. In fact, the system's structure is formed within the 3 discussed eras and can dynamically change. Second, it follows a concept called supply/demand. Third, it considers the 4 categories of resources. This consideration is respected at all of the system's structures. Fourth, to perform interpeer supply/demand, the system members use a concept called Region. Fifth, the system follows the local autonomy principle.

These are the features that make the peer-to-peer system different from any other distributed peer-to-peer system. We expect the Latin square to take into account all of the effects exerted by these 5 features when defining peer status.

3.6. Local supply/demand concept

In the peer-to-peer system's Jurassic era, any peer can enter or leave a region that has taken on the responsibility of responding to the global requests concerning a specific task. This behavior inside each peer results in a function called the SDL function. Based on this function, as a peer enters into the Jurassic era (due to the initialization of a global request at that peer or reception of a global request by the peer) and upon activation of a unit in the local peer, a unit called SDL is triggered. This unit aims to describe the peer status. It gets activated whenever the system switches from the local kernel into the system's kernel and it makes use of the data stored in Oasis to describe the peer's status as related to itself.

Counting the number of global requests in the SDOasis region, either of SL-type or DL-type, the unit can describe the peer's status concerning itself. SL requests refer to those requests initialized inside the peer and cannot be satisfied by the local operating system. DL requests refer to those requests accepted by the local peer and being executed on the peer as vice-processes. Of course, all of the global requests are accepted, if respecting the local autonomy of the peer.

$$\text{If}\, SL > DL\, \text{then}\, \text{Machine}_{State} = \text{Vendor}$$

$$\text{If}\, SL < DL\, \text{then}\, \text{Machine}_{State} = \text{Patron}$$

$$\text{If}\, SL = DL\ \text{ then }\ \text{Machine}_{State} = \text{Change to Kertaseh Era}$$

In the peer-to-peer system, the supply/demand function also determines if there should be a transition from one era to another. That is why defining a peer's status concerning itself is emphasized. On the other hand, the SDL unit specifies the peer's status regarding the requests that it cannot handle or the requests that it has received.

4. The way the Latin square is formed by LTE unit

The LTE unit is to manage the Latin square formation. This unit makes a Latin square within the Jurassic era. It is formed inside any peer to describe the peer's status regarding the global processes by which the peer gets affected somehow. Describing the peer's status in a given system, particularly complicated ones like distributed peer-to-peer systems, might not be conducted in an abstract manner. In other words, the peer's status concerning itself and the system cannot be described using the current processes being executed on the peer, the number of requests entered by the peer, and the number of requests the peer has forwarded.

In the LTE unit, an n $\times$ n data structure called the peer's Latin square is formed. This data structure is represented as LTE_{peer}. Thus, for each request sent or received by the peer, the Latin square is changed. There exists one Latin square for each type of resource. Requests related to a specific resource cause changes in the corresponding Latin square.

The Latin square is an n $\times$ n matrix. Each row in the matrix represents an activity. When the system's units meet a global request, regarding the operation, one single row of this matrix is initialized and is labeled

Activity$_x$ by the LTE. In other words, when the system replies to a request, its name is assigned to the first free row of the Latin square corresponding to the type of activity responded to.

The most important challenge of the LTE unit is the number of samples to be collected by this unit for the final formation of the Latin square. Therefore, to have each of the 4 types of resources of the Latin square properly formed, the LTE needs to gather data pertaining to some requests about each of the 4 types of resources. In theory, n may be any number, but for 3 reasons, this number has to be limited to a certain threshold. First, the LTE unit is run at the operating system's kernel level. Second, the nature of the requests selected to populate the LTE data structures should represent the peer's status concerning the executing global processes. The Latin square is formed to describe the peer's status regarding the global requests. The status is defined due to the peer's membership in or its exit from a specific region. Third, the information collection time should be much smaller compared to that of the system's average time of the common operations. On the other hand, the number of rows should not be taken too small; since it cannot define the operation taken for the execution of specific activities within the system well, taking the suitable value for n is largely dependent on factors such as the average speed of the system's member processors, average data transmission rate, and average bandwidth.

4.1. Method of obtaining the peer's Latin square

Next, we can describe the way that the Latin square may be created inside any local peer. In return for each occurrence of PBang X inside the local peer, the LTE unit is activated. Upon activation of the LTE, the processes' access to information is privileged, and for each request of the global process to resource X, the system call by which the resource is requested is investigated. Such systems calls constitute rows of the matrix. Its columns represent a number of executions. The element at the crossing of the calculated rows and columns indicates the workload of each activity. This value is calculated and put into its correct position by the LTE during the activity or the system call execution time. For each row, as long as the number of system calls executed equals n, upon reception of the global requests, the workload at the activity or system call execution time is assumed as the element at the crossing of the calculated row and column.

It should be noted that the LTE can manage completing rows and Latin squares (related to resource types) in parallel. The Table shows the schema of the Latin square form in each peer. It is worth mentioning that the Latin square obtained first shows the peer's status over a specific time interval. In other words, the Latin square is indicative of the peer's status as related to those global processes requesting resource X. As mentioned, a process is considered as a global one if it is forwarded because the local operating system has not been able to satisfy its resource requests. Thus, upon completion of the Latin square related to resource X, we may have a general scheme concerning the peer's status regarding resource X. Of course, this scheme is obtained considering the system effects.

Based on the results evaluated, and using the theory of probabilities and the huge number of instructions at the time when PBang X occurs, the probability of the workload caused by the execution of the mentioned set members to be identical goes to zero. That is one reason for using Latin squares.

4.2. Fundamental features of the Latin square

Paying attention to the way each a$_{ij}$ is calculated, it is realized that at the moment of the algorithm's execution, some instructions related to the formation of other parts of the peer-to-peer system inside the peer are in execution, and thus the workload is constantly changing.

Table 1. The Latin square formed inside of each peer.

Name of peer	Run_1	Run_2	....	Run_n
$\mathrm{Activity}_1$	A_{11}	A_{12}		A_{1n}
.	.		....	.
.	.		....	.
.	.		....	.
.	.		....	.
$\mathrm{Activity}_n$	An_1		....	A_{nn}

Therefore, the most important point about the peer's Latin square is that any number appears once in each row and column. This means that:

1. Different executions of i activity do not produce the same workload. In other words, each n times that activity i is executed, its workload never remains the same.

$$\forall\, \mathrm{j, i} \ni\ \mathrm{j} \gg 0\, \mathrm{and}\, \mathrm{j} \ll \mathrm{n} \rightarrow 2\mathrm{Simlar}\, \mathrm{a}_{ij} \tag{2}$$

2. Process behaviors and the operations they perform are not predictable and so many parameters affect them. Thus, at each certain moment, it is possible for new operations to be initiated inside the peers. Issues related to these operations, e.g., allocating their resources, directly affect the completion time of the global activity concerned.

At this point, we reach the conclusion that the assumptions set forth make us think that it is highly improbable to reach 2 identical a_{ij} elements in the same row. Actually, this probability is low and nearly equals zero.

The second point to be discussed about the peer's Latin square is that there is no identical number in the same column. This means that no 2 activities have the same execution time. It should be noted the statements made about the lack of similarity between the entries in a row could be extended to show that no 2 column entries are identical.

Next, taking into account the fact that 4 Latin squares exist inside of the peers and also that these are $\mathrm{n} \times \mathrm{n}$ matrices with no identical a_{ij} in each row and column, it may be stated that the peer's Latin square is in accordance with Latin square combinational design. Thus, in regard to the concept of the Latin square, we have:

$$\forall\, \mathrm{n} \geq 2\, \mathrm{and}\, \mathrm{Replace}\, \mathrm{n}\, \mathrm{as}\, 0 \tag{3}$$

4.3. Latin square updating time

The Latin square should describe the peer's status regarding the surrounding system. Therefore, the time needed for its formation and updating depends on the system's features. The peer's Latin square needs to be examined at 2 different times: first, the time when some changes occur at the system level in which the peer resides, and second, when the status of the local peer changes, such that the basic features defined and set for the peer change. In the Jurassic era, change in the system means change in the region in which the peer resides. Changing regions means that the peer is moving from one region to another, such that its Latin square must be recalculated. The second case occurs when the peer turns its status from vendor into patron. Under such conditions, the basic feature defined in the peer regarding the system has changed.

In a more general condition, when the frequency of changeability of each factor above exceeds the threshold set forth by the system's designer, the LTE can substitute them with the 2 following parameters. In this condition, changes in the system mean switching from the regions that respond to resource X to regions that respond to resource Y. However, X and Y will not be of the same value. Changes in the peer's status mean switching from Vendor_X into either Vendor_Y or Patron_X and from Patron_X into either Patron_Y or Vendor_X.

For each of the 2 rules discussed for the peer's status change, 1 of the 2 parameters used to examine the peer's status has changed. Thus, the output results from the peer status examination do not remain valid and the LTE must reevaluate the peer's status.

5. Evaluation

Several different evaluations can be performed, but to present an exact evaluation we have used the mentioned peer-to-peer system considering the workload under 2 different conditions. First, as seen in Figure 2, the peer-to-peer system contains a server and information collection, and the peer's status description is conducted on the basis of the 4-dimensional model already presented regarding the workload activity. Second, there is no server in the peer-to-peer system and using the 4-dimensional model based on the Latin square, the time required for gathering data, describing the peers' status regarding the workload activity, and the description given about the peer's individually are compared.

In the evaluation of the peer-to-peer system [29] in a server-oriented situation, we change the functionality of the coordinator peers at the system to the server and obtain the status of the peers based on the model and workload activity. Without any change in the system's status, we convert servers to ordinary peers to form a serverless system and use the Latin square model for describing the status of the peers. This test is repeated based on different measurements, such as the size of the system and the time for collecting the information 25 times.

In the former case, there is a concept, namely determining the system's status based on the data received from the system itself. The times required for gathering data using a server-oriented system and in a serverless system (a peer is randomly chosen to take action to collect the required data) are compared. On average, at each execution of the initial information-gathering model to describe peer status regarding the workload activity, the Latin square in the serverless architecture has been shown to be faster due to the small amounts of information that it requires.

In a server-oriented situation, increasing the size of the system has a direct relation with time. In test number 10, the size of the system is considered as half of that of test 11. As you can see in test 10, the server used 50 time units to describe the peers' status, since in the same situation by increasing the number of peers in test 11, the time increased to 97 time units. These tests are not done continuously, as we start the server in test 10 and obtain the time for collecting information, then shut down the server and increase the size of the system in test 11, and then again start the server to collect information and obtain the time. However, a serverless situation does not follow this behavior. As seen in Figure 2, although the system has doubled in size in test 11 compared to test 10, the time for collecting information has not doubled and the time for collecting information in test 10 is 23 time units, whereas in test 11 it is 26. This is because the collecting information process is only done on peers related to the global job with this peer.

In test 5, the time for collecting information in both situations, serverless and server-oriented, is the same. This is because the peers that start the global job in a serverless situation have used all of the peers in the system for running this job. Indeed, the peers' times for collecting information in tests 4 and 5 are approximately equal in the serverless situation, but the difference in the server-oriented situation is considerable. The difference of these 2 tests is only in the size of the system, where the size of the system in test 5 is smaller than that in test 4. These tests are done continuously, as in a server-oriented situation, at the moment of decreasing the size of the system to reach test 5, the server is in the system and is not down. By decreasing the size of the system, we expect that the time for collecting information decreases in a serverless situation significantly as well, but this does not happen. Due to the server-oriented situation, the server collects the information of all of the peers in the previous steps and has full information about the system since these tests are done continuously. With any new change in a peer or system, the server collects only the information of that part and updates the matrix of its Latin square. However, in a serverless situation with any new change, the peer collects the information of all of the peers related to it in running the job; therefore, the time for collecting information in this situation does not improve considerably.

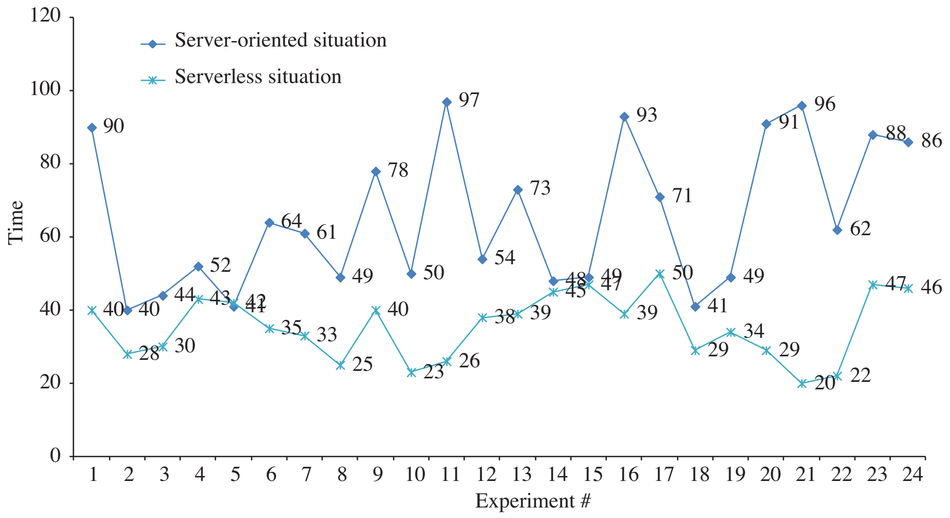

Figure 2. Time needed for collecting data to describe the peer's status.

Figure 3 specifies describing the peer's status based on 2 criteria. It is assumed that the workload on each peer in server-oriented systems is held, and Figure 3 presents an evaluation in which 20 peers have been examined at 2 different states. The green curve indicates the workload of the 20 peers while computational applications, MM5 [30] and WRF [31], are being executed. Peer number 2 is assumed to be the server. This peer is of full and adequate information about the workload of all of the other peers. To ignore the workload caused by the operating system (OS) operations, Linux Fedora Minimum 12 is installed on all of the peers. What peer number 2 calculates is the workload caused by a peer's participation in executing the global operations related to these applications.

The system is changed to investigate the Latin square in a serverless situation regarding the workload activity. While keeping its hardware and software untouched, some changes are made to the applications to meet the Latin square's requirements. Three reasons are given to show that the situation is kept the same as before. First, these changes are made only to remove the concept of the server peer(s). In such a condition, the initiating peer plays the role of the coordinator. Second, no change is made to the data structures responsible for executing the local operations, and so from the application's standpoint, no change is made to the hardware or software. Third, to keep both of the evaluations as similar as possible, it is assumed that when building the

Latin square, each peer only considers the process set as its generating set. Therefore, in both evaluations, the process is considered as the only resource based on which the workload is calculated. The red curve indicates the workload of each peer in the Kertaseh era, a stable situation in the peer-to-peer system. As can be seen, there are some differences between the workload calculated for the server-oriented system and the system using the Latin square in the serverless architecture. Generally, such differences are due to the communication overhead and traffic between the server and the other peers.

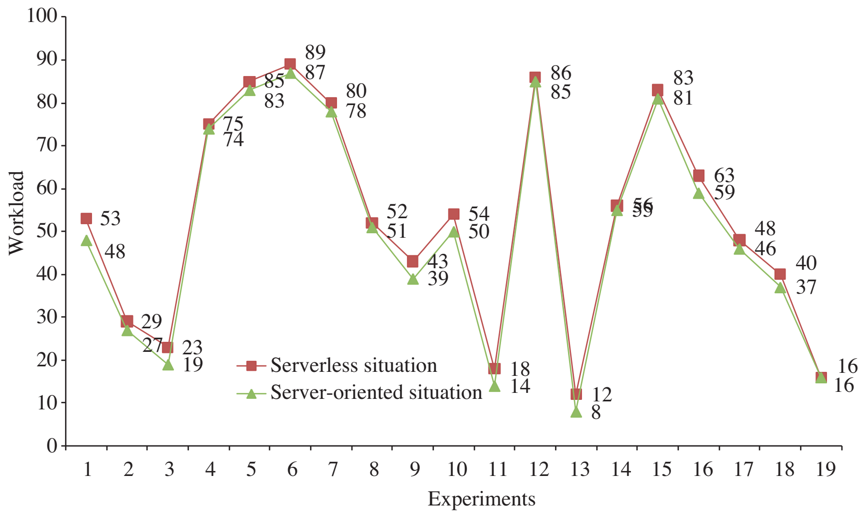

Figure 3. Peers' workload calculated in the server-oriented and serverless situations.

Figure 4 describes the workload calculated by the 2 architectures (server-oriented and serverless) with respect to time. As already shown in Figure 3, while the system is stable, there are some differences between the workload calculated in the 2 architectures. This will be discussed in detail in Section 7. In Figure 4, we are interested in investigating the way in which the workload is calculated by the 2 architectures. Peer number 5 in the evaluation given in Figure 3 has been considered. In the first test, peer number 5 is encased by a system in which peer number 2 is the server peer. This peer calculates the workload exerted on peer number 5, where the red curve in Figure 4 shows peer number 5's workload calculated by peer number 2 based on the status described for peer number 5 by the server. As we expected, at the beginning of the application, the execution of peer number 5's workload (the central processor's workload) fluctuates, but after 9 time units, we see that peer number 5's workload reaches a balance. The same test was conducted over peer number 5 when inside a system that made use of the Latin square to describe the peers' status in a serverless architecture.

The blue curve in Figure 4 represents the test. We see that, first, it takes longer to reach a balance when compared to the previous test, and, second, over the concerned time interval, it starts to fluctuate once again. The reasons behind such differences shall be explained as the challenge point of the Latin square in Section 6.

6. Discussion

As was seen in the evaluations, first, the volume of information needed for forming Latin squares inside the peers in a serverless architecture is very small compared to that needed in a server-oriented architecture. This was described implicitly in Section 1; due to the lack of a global view, the system member peers can collect less information in comparison with the server peers. This causes the size of the Latin square that is formed to be much smaller in the system in a serverless situation than in a server-oriented situation. In other words,

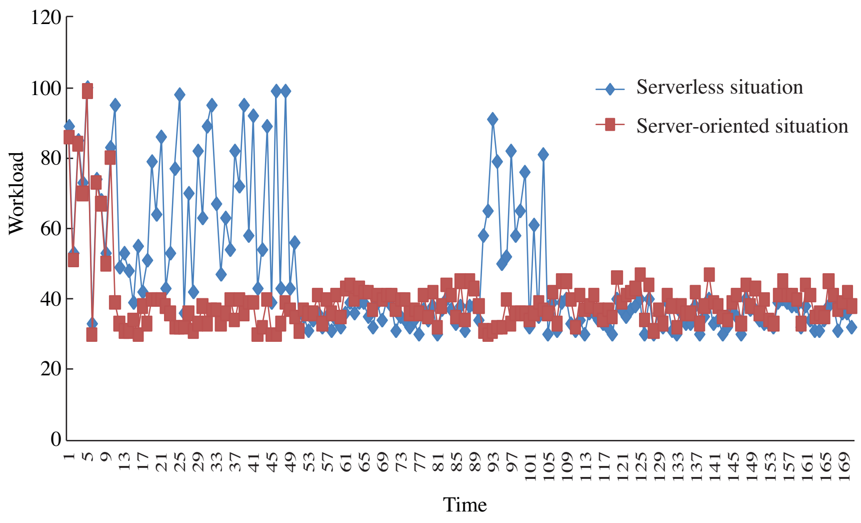

Figure 4. Calculated workload in serverless and server-oriented situations.

the workload required by peers who are members of the system for describing the peer's status is less than the workload needed by a server-oriented architecture.

Furthermore, as shown in the evaluations, any member in a serverless situation makes the same decision that a server makes, with the difference that the global operations to be conducted by the server to gather data are not necessary. On the other hand, the system's nature can be observed in the Latin squares formed in each peer. The system uses the 4-faceted resource-process model. This model is respected in any Latin square (and sets generating Latin squares) formed. Putting this all together, it means that the description of a peer's status using Latin squares matches the system's structure.

As we mentioned in Section 5, and in the evaluations shown in Section 6, the Latin square is unlike the traditional approaches used to describe the peers' status entering the distributed systems out of the server-oriented systems, as time is implicitly incorporated into the model. The existence of time in the generating sets and the high number of acceptable events used to describe the peer's status are all indicative of keeping track of time and the time information in Latin squares. On the other hand, the Latin square enables peers to make decisions about their own status without needing time synchronizations in a serverless architecture and they are only based on basic events.

As shown in the evaluations, the formation of Latin squares depends largely on the systems' eras. This means that the structure by which the decisions are made in the Jurassic era may not be similar to that of structures based on combinational designs used in the server. However, in the Kertaseh era, not only are all of the possible states generated, but also due to the small volume of Latin squares, describing the peers' status is much faster and more performable.

In the workload calculation evaluation (Figure 3), there is meaningful difference between the workloads calculated for the server-oriented situation compared to that of the Latin square model in a serverless situation. Performing the evaluations several times, we concluded that the difference is always constant when the evaluations are repeated so many times. Let the constant be C, and then for the sake of finding out why such a constant ever exists inside the systems concerned, we need monitoring of the central processor's workload. When the behavior of the central processor when the system is server-oriented is studied, then it is understood that while the central processor is busy running applications, some time is spent responding to the requests sent by the server peer. To be more exact, in both of the models, the operating system obtains the central processor's control. Since in both systems the minimum operating system is used, the time spent should be small enough to be neglected. On the other hand, when the server peer asks for the status, the operating system gets activated, and by checking the existing workload at the central processor, it measures it and sends the information to the server. However, in a serverless situation, based on the 4-directional model, information is stored at a local peer. In other words, calculating the workload based on the local information and not using a server to have it calculated causes the difference.

As we noted in Sections 5 and 6, in the serverless situation, based on the 4-dimensional model, more time is needed to describe the peers' status. The first question about Figure 3 may be why stability is of great concern. This condition is the one in which we desire to execute applications. The condition includes times, where we may easily make decisions about the parameters affecting the system's status. Fluctuating parameters definitely affect the system's status during nonstable conditions, such that the peers' status described by any model cannot be accurate.

If we regard peer number 5's stability in Figure 4, we notice that in the server-oriented situation, peer number 2 (as the server) sees peer 5's instability period as much smaller. That is because of requesting the peer's status and processes being initiated in peer number 5. The time above may be neglected, reasoning that this time is spent to describe peer number 5's status through its living at the stable condition. However, this does not hold for the 4-dimensional model. In order to have it studied, we should examine peer number 5's status in the system. As already discussed in Sections 2 and 3, Latin square combinational designs that describe the status of the peers take into account an important factor that is called the system's effects (or parameters). In Figure 4, the blue curve related to the description of peer number 5's status using the 4-dimensional model is shown regarding the system's effects. If we take a second look at the primary period spent on executing the applications inside peer 5, we realize that its description on the basis of the 4-dimensional model has higher fluctuations when compared to peer number 5's description on the basis of the server model. The reason here is actually related to the systems' nature. When applications inside peer number 5 are getting activated, the 4-dimensional model describes the status of peer 5 in the Mosaic era. Upon the occurrence of the first PBang process inside peer number 5, the 4-dimensional model describes the status of the peer using the Latin square's combinational designs' structures in the Jurassic era. The most important feature of the Jurassic era in the system is the lack of stability. This means that peer number 5 is fluctuating (entering and leaving) in the processing regions of the distributed peer-to-peer system. Thus, in the simplest way, it is concluded that due to changes in the system's effects, the peer's status description turns variable, as well. In the Jurassic era, the Latin square is directly affected by 2 sets, SAtrib and MAtrib. On the other hand, these 2 sets change in accordance with changes taking place regarding the peer's status inside the system. After a certain period, we see that the peer's description using the Latin square enters the Kertaseh era. The most important feature of the Kertaseh era in the peer-to-peer system is the dominant stability across the system. The 4-dimensional model shows a stable and balanced behavior during the Kertaseh era.

In this evaluation, we see another event of PBang within the Kertaseh era. This means that a processing request has been encountered inside peer number 5 that neither the peer itself nor any of the peers within the same region in which peer 5 resides have not been able to satisfy. Thus, peer 5 has entered the secondary Jurassic era.

One of the main differences of the 4-dimensional model compared to others is its wide support for resources. Performing the evaluations above regarding any other resource than the process causes the server-oriented architecture to fail. This is because either the frequency of the changes are too high, such that the server cannot provide a description of the peer's status, or some information sent by the peers is meaningless to the server peer. The second reason is more often seen in the case of special types of resources.

Using the 4-dimensional model in a serverless architecture, we benefit from the capability of describing the peer's status: first, in regard to the peer-to-peer system, getting the independence of the central structures such as the server-oriented architecture. Second, the peer's status will not be described just based on the abstract status of the peer. In other words, the system considers the peer in a system and the effects of that system. Third, it is compatible with the distributed peer-to-peer systems lacking certain stability periods given by the server-oriented systems.

The 4-dimensional model in a serverless situation only has a challenge when the peer starts a global job and uses most of the peers in the system in running it. In this situation, with any change in the parameters of the model, the starter peer collects the information of the numerous peers and updates their describing matrixes; this is because the peer has no total overview of all of the parameters of the model and it is a time-consuming process. However, in a server-oriented situation, the server has complete information about all of the parameters of the model (MAtrib, SAtrib, AAtrib), and by changing any of these parameters, the server can detect the change and update the related information.

7. Conclusion

In this paper, a 4-dimensional model is proposed to describe the peers' status in a server-oriented architecture. The 4-dimensional model is also considered for the peer-to-peer architecture to overcome the limitations of such systems in describing the status of the peers. A Latin square is applied to the distributed peer-to-peer system. In this peer-to-peer system, both architectures have been compared. The proposed model requires minimal data to function. By considering the sets MAtrib, AAtrib, the 2 concepts of time, and the system's features, it is possible to have the peer's status described by first using the local data of each peer, then taking into account the status of all of the peer's resources, and finally considering the peer-to-peer system as a whole. Using a 4-dimensional model in a serverless architecture, as opposed to a server-oriented architecture, increases the precision of the description given by the peer's status of any basic activity like the workload. However, there are 2 main challenges about the descriptions obtained using a 4-dimensional model in a serverless architecture. First, this model requires the 'system's effect factors' set, which needs proper knowledge about the system and those parameters affecting the peer's status. Second, in a serverless architecture, due to the dependency of the 4-dimensional model on the underlying system, it needs more time to reach stability compared to a server-oriented architecture. In addition, unlike the server-oriented architecture, after reaching stability, due to the stochastic nature of the system, no guarantee is given on a sustainable stability. Despite a server-oriented architecture, the 4-dimensional model is in accordance with the nature of distributed systems and satisfies many of its requirements, such as there being no need for global information. In addition, unlike the server-oriented architecture, different resource types and times are taken into account when describing the peer's status. The 4-dimensional model in the peer-to-peer system provides an opportunity to the local operating system to make use of its own structures in the same way as it does when using kernel structures. Using these structures, an OS can better decide about the peer's status and the system encompassing it. This feature enables existing processes within the peer to decide about the global processes. In other words, decisions about acceptance or rejection of global process are made more accurately using the information stored in the Latin square. This important aspect will be further investigated in our future work.

References

[1] R. Ranjan, A. Harwood, R. Buyya, "Peer-to-peer based resource discovery in global grids: a tutorial", IEEE Communications Surveys and Tutorials, Vol. 10, pp. 6–33, 2008.

[2] D. Grosu, A.T. Chronopoulos, "Noncooperative load balancing in distributed systems", Journal of Parallel and Distributed Computing, Vol. 65, pp. 1022–1034, 2005.

[3] S. Viswanathan, B. Veeravalli, T.G. Robertazzi, "Resource-aware distributed scheduling strategies for large-scale computational cluster/grid systems", IEEE Transactions on Parallel and Distributed Systems, Vol. 18, pp. 1450–1461, 2007.

[4] A. Iamnitchi, I. Foster, "On fully decentralized resource discovery in grid environments", Computer Science, Vol. 2242, pp. 51–62, 2001.

[5] D.P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, D. Werthimer, "SETI@home: an experiment in public-resource computing", Communications of the ACM, Vol. 45, pp. 56–61, 2002.

[6] Avaki Corporation, AVAKI Grid Software: Concepts and Architecture, 2002.

[7] C. Roncancio, M.P. Villamil, C. Labbé, P. Serrano-Alvarado, "Data sharing in DHT based P2P systems", Transactions on Large-Scale Data- and Knowledge-Centered Systems, Vol. 1, pp. 327–352, 2009.

[8] R. Huebsch, B. Chun, J. Hellerstein, B. Loo, P. Maniatis, T. Roscoe, S. Shenker, I. Stoica, A. Yumerefendi, "The architecture of PIER: an internet-scale query processor", International Conference on Innovative Data Systems Research, pp. 28–43, 2005.

[9] M. Ripeanu, I. Foster, A. Iamnitchi, "Mapping the Gnutella network: properties of large-scale peer-to-peer systems and implications for system design", Journal of IEEE Internet Computing, Vol. 6, pp. 50–57, 2002.

[10] I. Clarke, O. Sandberg, B. Wiley, T.W. Hong, "Freenet: a distributed anonymous information storage and retrieval system", Workshop on Design Issues in Anonymity and Unobservability, pp. 46–66, 2000.

[11] G. Bolcer, "Magi: architecture for mobile and disconnected work?ow", IEEE Internet Computing, Vol. 4, pp. 46–54, 2000.

[12] P. Stanhope, Get in the Groove: Building Tools and Peer-To-Peer Solutions with the Groove Platform, New York, Wiley, 2002.

[13] H. Wang, J. Liu, K. Xu, "Measurement and enhancement of BitTorrent-based video file swarming", Peer-to-Peer Networking and Applications, Vol. 3, pp. 237–253, 2010.

[14] C. Kaleli, H. Polat, "P2P collaborative filtering with privacy", Turkish Journal of Electrical Engineering & Computer Sciences, Vol. 8, pp. 101–116, 2010.

[15] L. Huang, M. Garofalakis, A.D. Joseph, N. Taft, "Approximate decision making in large-scale distributed systems", NIPS Workshop: Statistical Learning Techniques for Solving Systems Problems, 2007.

[16] D.S. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, Z. Xu, "Peer-to-peer computing", HP Laboratories, Palo Alto, CA, USA, Technical Report, 2003.

[17] E. Anderson, D. Patterson, "Extensible, scalable monitoring for clusters of computers", The 11th USENIX Conference on System Administration, 1997.

[18] F. Zhu, M.W Mutka, L.M. Ni, "Service discovery in pervasive computing environments", IEEE Pervasive Computing, Vol. 4, pp. 81–90, 2005.

[19] X. Zheng, V.R. Oleshchuk, "A survey on peer-to-peer SIP based communication systems", Peer-to-Peer Networking and Applications, Vol. 3, pp. 257–264, 2010.

[20] D.R. Kumari, H. Guclu, M.T. Yuksel, "Ad-hoc limited scale-free models for unstructured peer-to-peer networks", Peer-to-Peer Networking and Applications, Vol. 4, pp. 92–105, 2008.

[21] E. Meshkova, J. Riihijärvi, M. Petrova, P. Mähönen, "A survey on resource discovery mechanisms, peer-to-peer and service discovery frameworks", Computer Networks: The International Journal of Computer and Telecommunications Networking, Vol. 52, pp. 2097–2128, 2008.

[22] A. Iamnitchi, I. Foster, D.C. Nurmi, "A peer-to-peer approach to resource discovery in grid environments", 10th IEEE International Symposium on High Performance Distributed Computing, 2002.

[23] J.R. Albrecht, D.L. Oppenheimer, A. Vahdat, D.A. Patterson, "Design and implementation trade-offs for wide-area resource discovery", ACM Transactions on Internet Technology, Vol. 8, pp. 1–44, 2008.

[24] I. Foster, C. Kesselman, "Globus: a metacomputing infrastructure toolkit", International Journal of Supercomputer Applications, Vol. 11, pp. 115–128, 1996.

[25] C. Doulkeridis, A. Vlachou, K. Nørvåg, Y. Kotidis, M. Vazirgiannis, "Efficient search based on content similarity over self-organizing P2P networks", Peer-to-Peer Networking and Applications, Vol. 3, pp. 67–79, 2010.

[26] A. Barak, O. La'adan, "The MOSIX multicomputer operating system for high performance cluster computing", Future Generation Computer Systems, Vol. 13, pp. 361–372, 1998.

[27] R. Raman, M. Livny, M. Solomon, "Matchmaking: distributed resource management for high throughput computing", High Performance Distributed Computing, pp. 140–146, 2002.

[28] C.F. Laywine, Discrete Mathematics Using Latin Squares, Hoboken, NJ, USA, Wiley Interscience, 1998.

[29] M. Sharifi, S.L. Mirtaheri, E. Mousavi Khaneghah, "A dynamic framework for integrated management of all types of resources in P2P systems", Journal of Supercomputing, Vol. 52, pp. 149–170, 2010.

[30] R. Noronha, D.K. Panda, "Performance evaluation of MM5 on clusters with modern interconnects: scalability and impact", Proceedings of the 11th International Euro-Par Conference on Parallel Processing, pp. 134–145, 2005.

[31] D. Morton, O. Nudson, D. Bahls, P. Johnsen, D. Arnold, I. Schicker, "Grand-scale WRF testing on the Cray XT5 and XE6", Cray User Group Proceedings, 2011.