UDC 004.75

GRID-INFRASTRUCTURE SIMULATION[1]

A.Ju. Shelestov, N.N. Kussul, S.V. Skakun

Space Research Institute NASU-NSAU, Kyiv, 03680, prosp. Glushkova 40,
+38044-526-25-53, inform@ikd.kiev.ua

In a paper simulation modelling of Grid infrastructure is carried out in order to investigate and analyze its productivity. For this purpose, GridSim modelling package is used. An approach based on the operational analysis is applied for the performance analysis of the Grid system. Such approach enables qualitative and quantitative estimation of the Grid system performance.

Introduction

Grid systems are becoming standard solutions for enabling remote computations execution and distributed data access and processing in environments of the different level of scalability. As it was stated by originator of Grid Ian Foster "The real and specific problem that underlies the Grid concept is coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations.... This sharing is, necessarily, highly controlled, with resource providers and consumers defining clearly and carefully just what is shared, who is allowed to share, and the conditions under which sharing occurs. A set of individuals and/or institutions defined by such sharing rules form what we call a virtual organization (VO)" [**Ошибка! Источник ссылки не найден.**].

The task of Grid system could be formulated as connecting data, processing powers and algorithms that distributed over the network for solving particular problems. Grid system should be universal up to some degree, so these problems should not be hardcoded during its development. Instead, a set of problems being solved in a Grid environment must be open for modifications and addition of new problems. This goal is achieved by introducing standard interfaces for communicating between different kinds of Grid resources and clients.

Space agencies all over the world have started working on evaluation of Grid technology in their application areas. This is due to the fact that Earth observation (EO) domain is characterized by the acquisition of large amounts of data from satellites. Furthermore, the single EO product as is defined and the data after its initial processing may easily exceed the gigabyte size. Thus, problems of storing, indexing for quick retrieval on application's demand as well as distributed computing arise within aforementioned area. Grid technology can provide needed solution for this problem.

Since Grid deals with geographically distributed resources it introduces a number of resource management and application scheduling challenges. The resource management and scheduling systems for Grid computing need to manage resources and application execution depending on either resource consumers' or owners' requirements, and continuously adapt to changes in resource availability. The management of resources and scheduling of applications in such large-scale distributed systems is a complex undertaking. In order to prove the effectiveness of data processing simulation should be performed. In this paper an approach based on simulation tools and operational analysis is used to model Grid system.

1. Overview of Grid Systems

Nowadays Grid technology is widely applied for the solutions of various problems in many domains [**Ошибка! Источник ссылки не найден.**]. These applications span a wide spectrum. In this section we give a brief overview of Grid systems that are used for satellite data processing.

European DataGrid Project (EDG) [**Ошибка! Источник ссылки не найден.**] that was funded by EC begun in 2001. It was one of the first Grid-enabled projects allowing ESA to gain firsthand experience in the use of emerging Grid technologies. To test the capabilities of the system, it was decided to use data from ERS-2's GOME [**Ошибка! Источник ссылки не найден.**] instrument, consisting of global atmospheric-ozone measurements collected over several years of the mission. This instrument generates some 400 terabytes of data products per year that have to be catalogued, archived and processed.

Japan Aerospace eXploration Agency (JAXA) [**Ошибка! Источник ссылки не найден.**] and KEIO University started establishing "Digital Asia" system aimed at semi-real time data processing and analyzing. They use GRID environment to accumulate knowledge and know-how to process remote sensing data. The problems of radiometric rectification and composition of remotely sensed data are concerned.

National Aeronautics and Space Administration (NASA) have created Information Power Grid (IPG) [**Ошибка! Источник ссылки не найден.**] targeting an operational Grid environment incorporating major computing

[1] The work is supported by STCU and NASU grant "GRID technologies for environmental monitoring using satellite data" (project # 3872).

and data resources at multiple NASA sites in order to provide an infrastructure capable of routinely addressing larger scale, more diverse, and more transient problems than is possible today. Nowadays IPG have approximately 600 CPU nodes of Computing resources and 30-100 Terabytes of archival information/data storage resources.

2. Grid testbed description

The logical structure of the Grid testbed depicted on Fig. 1 consists of 6 resources including a high performance cluster (Cluster of Institute of Cybernetics National Academy of Sciences of Ukraine, IC NASU), as well as 4 third-party grid nodes (NYX Grid node, Masquerade Grid node, IKD Grid node, Altair Grid node). The physical structure of the Grid testbed depicted on Fig. 2. These resources are connected via wide area network Internet. The specific parameters of each resource are listed below.

— Cluster of IC NASU: x86, CPU 24x2x2.67 MHz, RAM 1 GB.

— NYX Grid node: x86 Gentoo Linux box, CPU 1.3 Mhz, RAM 1 GB, HDD 40 GB, deployed Globus Toolkit 4.0 (GT).

— Masquerade Grid node: AMD64 Gentoo Linux box, CPU 2.2 Mhz, RAM 1 GB, HDD 160 GB, deployed

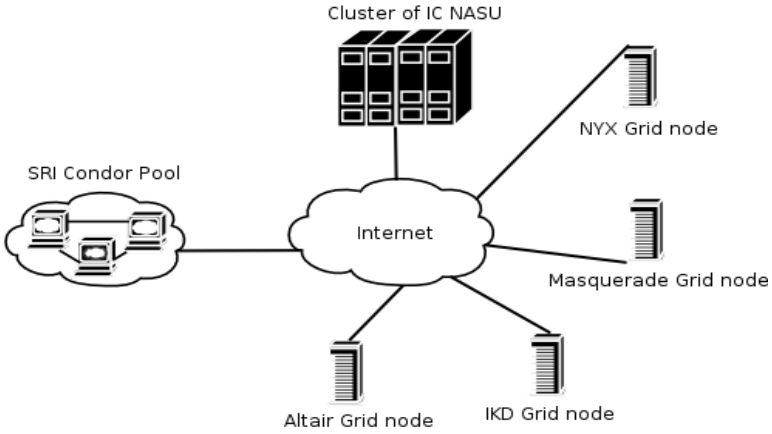

Fig. 1. Logical structure of the Grid testbed

GT 4.0.

— IKD Grid node: x86 FreeBSD box, CPU 1.3 Mhz, RAM 1 GB, HDD 40 GB, deployed Globus Toolkit 4.0

— Altair Grid node: x86 Gentoo Linux box, CPU 1.5 Mhz, RAM 512 GB, HDD 60 GB, deployed Globus Toolkit 4.0.

The logical structure of the Grid testbed presented above is simplified for convenience. Physical structure of the Grid testbed (Fig. 2) is given for demonstration the interconnection of the resources (Cluster of IC NASU and NYX, Masquerade, IKD Altair Grid nodes) via network and specifying the situation of 5 routers (SRI Router, Global Router,

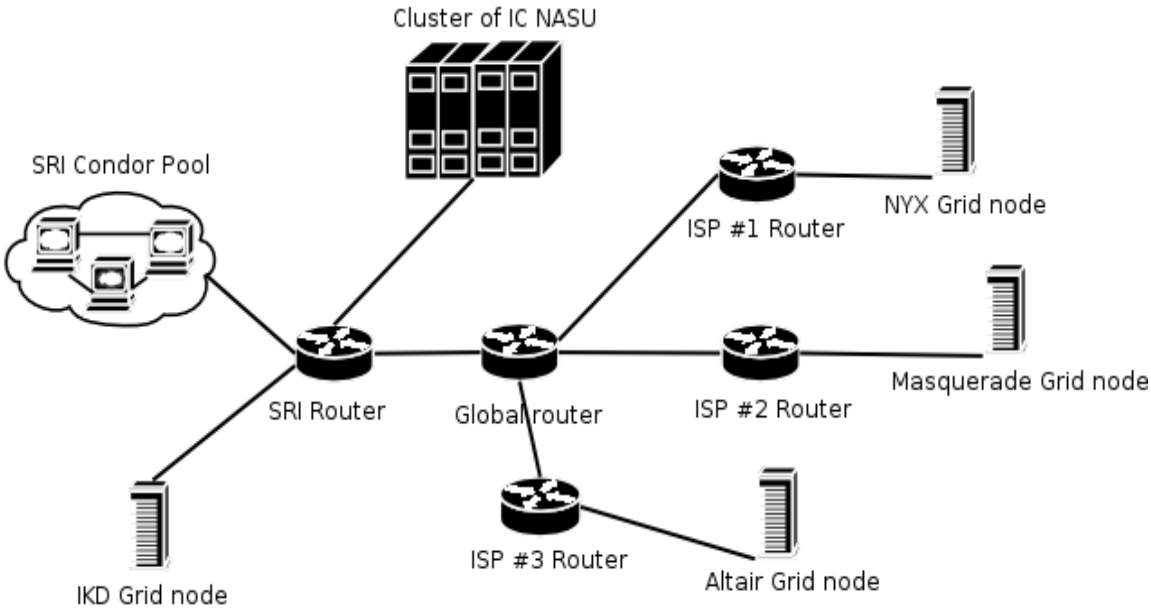

Fig. 2. Physical structure of the Grid testbed

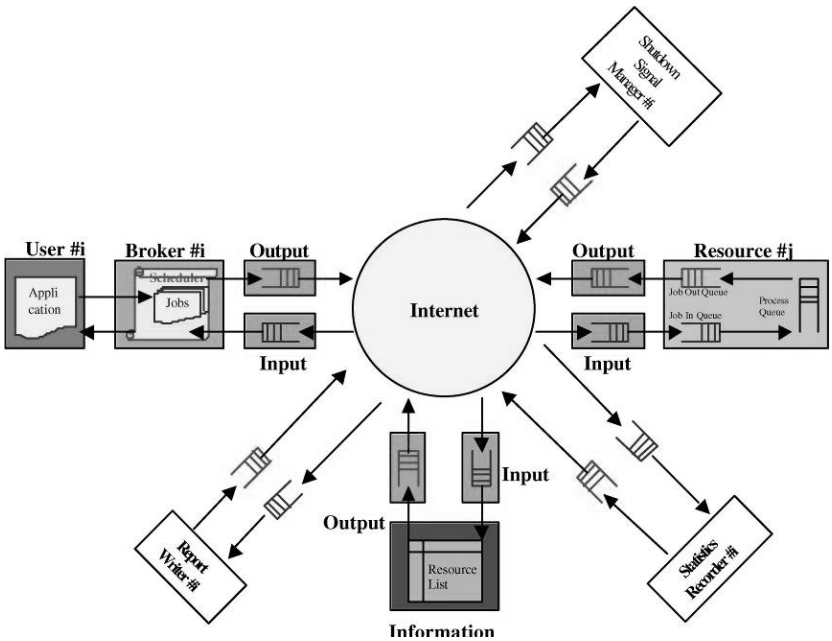

Fig. 3 A flow diagram for GridSim based simulations

ISP #1 Router, ISP #2 Router, ISP #3 Router).

3. Gridsim-based model of the testbed

A generic flow diagram for GridSim [**Ошибка! Источник ссылки не найден.**] based simulations is depicted on Fig. 3. The model used in our evaluation consists of a Grid Information Service (GIS), broker and one or more resources and users. The functionality of each component is briefly described below.

Grid entities model. GridSim supports entities for simulation of single processor and multiprocessor, heterogeneous resources that can be configured as time-or space-shared systems. It allows setting of the clock to different time zones to simulate geographic distribution of resources. It supports entities that simulate networks used for communication among resources. During simulation, GridSim creates a number of multi-threaded entities, each of which runs in parallel in its own thread. An entity's behaviour needs to be simulated within its body() method, as dictated by SimJava.

A simulation environment needs to abstract all the entities and their time-dependent interactions in the real system. It needs to support the creation of user-defined time-dependent response functions for the interacting entities. The response function can be a function of the past, current, or both states of entities. GridSim based simulations contain entities for the users, brokers, resources, information service, statistics, and network based I/O. The design and implementation issues of these GridSim entities are discussed below.

User. Each instance of the User entity represents a Grid user. Each user may differ from the rest of users with respect to the following characteristics:

— types of job created, e.g. job execution time, number of parametric replications, etc.;

— scheduling optimization strategy, e.g. minimization of cost, time, or both;

— activity rate, e.g. how often it creates new job;

— time zone; and

— absolute deadline and budget; or

D-and B-factors, deadline and budget relaxation parameters, measured in the range [0, 1] express deadline and budget affordability of the user relative to the application processing requirements and available resources.

Broker. Each user is connected to an instance of the Broker entity. Every job of a user is the first submitted to its broker and the broker then schedules the parametric tasks according to the user's scheduling policy. Before scheduling the tasks, the broker dynamically gets a list of available resources from the global directory entity. Every broker tries to optimize the policy of its user and therefore, brokers are expected to face extreme competition while gaining access to resources. The scheduling algorithms used by the brokers must be highly adaptable to the market's supply and demand situation.

Resource. Each instance of the Resource entity represents a Grid resource. Each resource may differ from the rest of the resources with respect to the following characteristics:

— number of processors;

— cost of processing;

— speed of processing;

—internal process scheduling policy, e.g. time-shared or space-shared;

— local load factor; and

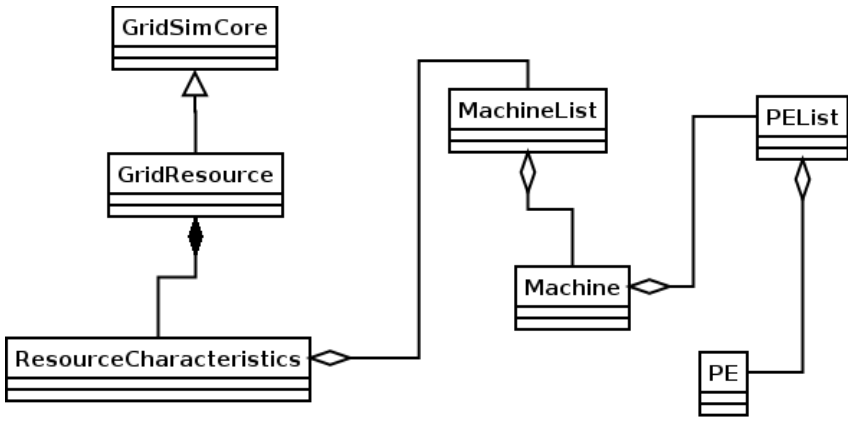

Fig. 4. Class diagram of a GridResource model

— time zone.

GridResource extends the GridSimCore class for gaining communication and concurrent entity capabilities. An instance of this class stimulates a resource with properties defined in an object of ResourceCharacteristics class Fig. 4.

The process of creating a Grid resource is as follows:

A resource having a single machine with one or more PEs (Processing Elements) is managed as a time-shared system using a round-robin scheduling algorithm. A resource with multiple machines is treated as a distributed memory cluster and is managed as a space-shared system using FCFS (First Come First Serve) scheduling policy or its variants.

The resource speed and the job execution time can be defined in terms of the ratings of standard benchmarks such as MIPS and SPEC. They can also be defined with respect to the standard machine. Upon obtaining the resource contact details from the Grid information service, brokers can query resources directly for their static and dynamic properties.

Grid information service. Each VO domain has a GIS that is responsible for providing resource registration services and keeping track of a list of resources available in the Grid. The information given by a resource consists of a name, location, hardware specification. Users contact their GIS if they want to know the location and availability of local resources. The brokers can query this for resource contact, configuration, and status information.

Input and output. The flow of information among the GridSim entities happens via their Input and Output entities. Every networked GridSim entity has I/O channels or ports, which are used for establishing a link between the entity and its own Input and Output entities. Note that the GridSim entity and its Input and Output entities are threaded entities, i.e. they have their own execution thread with body() method that handles events. The use of separate entities for input and output enables a networked entity to model full duplex and multi-user parallel communications. The support for buffered input and output channels associated with every GridSim entity provides a simple mechanism for an entity to communicate with other entities and at the same time enables modelling of the necessary communications delay transparently.

Testbed topology model. The topology model used in this work assumes that processing resources and user entities are interconnected into WAN (wide area network) network. In other words, the network has a simple topology in which two nodes are connected by a small sequence of links that form an end-to-end connection. There are routers connecting segments of the network. Data transfer is assumed to be fragmented into packets (jobs, Gridlets) and pipelined through the links that form a route. Thus, the general behaviour is that data transfers happen at the speed of the bottleneck link. This model is inspired on the flow of fluids in pipes and has been shown to be a good approximation, thus, creating, in the context of grid simulation, an appropriate alternative to the use of more established complex packet level models.

In this work the testbed topology model was built using GridSim. The class in which this topology is implemented is named GridTestbed.

Within this class the testbed topology model is implemented with the help of GridTestbed objects: 5 gridResources (IKD, Altair, Nyx, Masquerade, Cluster), 5 routers (globalRouter, router_Masquerade, router_IKD, router_Nyx, router_Altair) and links between them.

The process of the creation of the GridTestbed objects takes place in the following order. First of all the Grid resources are created and their characteristics are specified (Listing 1).

```
GridResource gridResource1 = createGridResource("IKD",1,1,0,500,"x86","FreeBSD");
GridResource gridResource2 = createGridResource("Altair",1,1,0,474,"x86","Linux");
GridResource gridResource3 = createGridResource("Nyx",1,1,0,474,"x86","Linux");
GridResource gridResource4 = createGridResource("Masquerade",1,1,0,1623,"AMD64","Linux");
GridResource gridResource5 = createGridResource("Cluster",24,2,1,988,"x86","Linux");
```

Listing 1 Creation of Grid resources

As a result we obtain the following set of resources (Listing 2).

```
Name : IKD       Architecture : x86    OS : FreeBSD MIPS : 500   ID : 6
Name : Altair    Architecture : x86    OS : Linux   MIPS : 474   ID : 11
Name : Nyx       Architecture : x86    OS : Linux   MIPS : 474   ID : 16
Name : Masquerade Architecture : AMD64  OS : Linux   MIPS : 1623  ID : 21
Name : Cluster   Architecture : x86    OS : Linux   MIPS : 988   ID : 26
```

Listing 2 List of created Grid resources and their characteristics

Then creation of routers and links takes place. The number of classes, objects and corresponding methods provided by GridSim where used to create the routers and links in the network topology model of this work. The routers are connected to resources and users via links. This is done according to the following routing tables (Listing 3).

```
--- Routing Table for globalRouter ---
router_MasqueraderGlobal_rMasquerade_link
router_IKD              rGlobal_rIKD_link
router_Nyx              rGlobal_rNyx_link
router_Altair           rGlobal_rAltair_link
---------------------------------------

--- Routing Table for router_IKD ---
User_4          User_4_link
IKD             IKD_link
User_3          User_3_link
User_2          User_2_link
User_1          User_1_link
globalRouter    rGlobal_rIKD_link
User_0          User_0_link
Cluster         Cluster_link
---------------------------------------

--- Routing Table for router_Altair ---
Altair          Altair_link
globalRouter    rGlobal_rAltair_link
---------------------------------------

--- Routing Table for router_Nyx ---
Nyx             Nyx_link
globalRouter    rGlobal_rNyx_link
---------------------------------------

--- Routing Table for router_Masquerade ---
globalRouter            rGlobal_rMasquerade_link
Masquerade              Masquerade_link
---------------------------------------
```

Listing 3 Routing tables

The workload model. The workload model in the terms of the chosen GridSim simulation package is described using the following parameters:

- Number of users in the system Nusers;
- Number of Gridlets for each user Ngridlets;
- Distribution of job submission time delay for each user;
- Time (delay) between two successively submitted Gridlets for each specified user (think time) Zi;

The product Nusers * Ngridlets constitutes to the total number of jobs in the system (in our case the system is considered to be a closed one).

In GridSim, each independent task may require varying processing time and input files size. Such tasks are created and their requirements are defined through Gridlet objects. A Gridlet is a package that contains all the information related to the job and its execution management details such as job length expressed in MIPS, disk I/O operations, the size of input and output files, and the job originator. These basic parameters help in determining execution time, the time required to transport input and output files between users and remote resources, and returning the processed Gridlets back to the originator along with the results. The GridSim toolkit supports a wide range of Gridlet management protocols and services that allow schedulers to map a Gridlet to a resource and manage it throughout the life cycle.

Gridlet attributes, which model the jobs run on the resources of the testbed polygon, can also be considered as parameters of the workload.

```
Gridlet(int gridletID,
        double gridletLength,
        long gridletFileSize,
        long gridletOutputSize)
```

where gridletID - the unique ID of this Gridlet; gridletLength - the length or size (in MI) of this Gridlet to be executed in a GridResource; gridletFileSize - the file size (in byte) of this Gridlet BEFORE submitting to a GridResource; gridletOutputSize - the file size (in byte) of this Gridlet AFTER finish executing by a GridResource.

Experiment's scenarios description & implementation. On the constructed testbed model performance evaluation experiments can be carried out according to various scenarios, let us consider three of them:

Experiment Scenario #1. User-centric scheduling with random selection of the target resources.

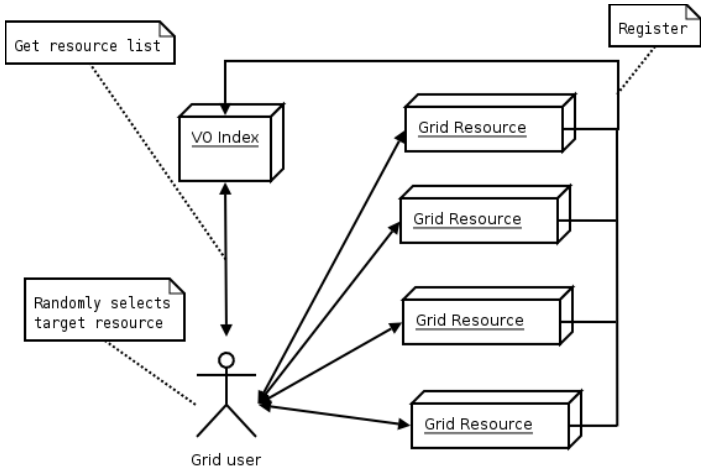

Fig. 5 User-centric scheduling with random selection of the target resources

Figure 5 shows the interaction between relevant components in our model according to the first scenario. The explanation of these interaction steps are explained below:

1. Grid User specifies his jobs and queries a list of available resources to the GIS (VO Index) as each resource advertises its availability to the designated GIS (VO Index)

2. Grid User 0 randomly selects the resource and sends his/her jobs (in the form of Gridlets) with specified parameters to the chosen resource. The same applies to User 1 and all the other Users, taking into account the distribution of job submission time delay for each user.

3. After the job has been run on the resource it is returned to the corresponding user, who in turn sends the next job according to the same scenario as described above.

Experiment Scenario #2. User-centric scheduling with selection of the target resources on the basis of dynamic information.

Experiment Scenario #3. Usage of standalone Grid broker with selection of the target resources on the basis of dynamic information.

After performing a detailed study of each of these scenarios a conclusion was derived that the second and third scenarios are of great complexity, and their realization prompts the appearance of a multitude of secondary scientific problems, the solution of which goes far beyond the boundaries of this work. Therefore further discussion will be concentrated on the implementation of the first scenario only.

Experiment data analysis and results

GridSim simulation results. As the result of modelling carried out using GridSim modelling package according to the “User-centric scheduling with random selection of the target resources” experiment scenario numerous data was obtained. This data after adequate analysis allows us to make performance analysis of the modelled system.

After each iteration of our experiment report and statistic files are generated. The information contained in these files includes resource model description, network topology model data, data regarding users and their jobs, history and statistics of job execution on resources. The output information about the modelled computational resources is presented in the following form given in Listing 4:

```
Name : IKD        Architecture : x86 OS : FreeBSD    MIPS : 500      ID : 6
Name : Altair     Architecture : x86      OS : Linux MIPS : 474      ID : 11
Name : Nyx        Architecture : x86      OS : Linux MIPS : 474      ID : 16
Name : Masquerade Architecture : AMD64    OS : Linux MIPS : 1623     ID : 21
```

```
Name : Cluster    Architecture : x86      OS : Linux MIPS : 988      ID : 26
```

Listing 4 Resource model data

The output information about the system's users is presented in the following form given below in Listing 5:

```
Creating a grid user entity with name = User_0, and id = 41
User_0: Creating 2 Gridlets
Creating a grid user entity with name = User_1, and id = 45
User_1: Creating 2 Gridlets
Creating a grid user entity with name = User_2, and id = 49
User_2: Creating 2 Gridlets
```

Listing 5 Example of output data about users of the modelled system

The information about network topology of our system is provided in the form of output routing tables of the modeled system's routers. The example of such table is given in the Listing 6 below:

```
--- Routing Table for router_IKD ---
User_4          User_4_link
User_1          User_1_link
User_8          User_8_link
Cluster         Cluster_link
User_5          User_5_link
User_2          User_2_link
globalRouter            rGlobal_rIKD_link
IKD             IKD_link
-------------------------------------
```

Listing 6 Example of output routing table for router_IKD router

As the result of the modelling process for each user specified in the model of our Grid system a report file is generated containing the statistics of job execution. In the example given below, in Listing 4 it can be seen that the User_9 user submitted for execution two Gridlets (jobs). The first job (gridlet #0) was successfully executed on Altair resource (ID: 11), while the second job (gridlet #0) was successfully executed on Nyx resource (ID: 16). In the Cost column the job's cost is presented. In our case the cost of one second of execution is the same for all resources and equals 1G$ (Grid-dollar, standard unit). Further in the Listing 7 we see the time axis (model time) with detailed stages of user's jobs execution.

```
============== OUTPUT for User_9 ==========
Gridlet ID    STATUS    Resource ID    Cost
   0        Success        11        21.722046413502106
   1        Success        16        11.29852320675107
Time below denotes the simulation time.
Time (sec)       Description Gridlet #0
-----------------------------------------
0.00   Creates Gridlet ID #0
0.00   Assigns the Gridlet to User_9 (ID #77)
58.962   Allocates this Gridlet to Altair (ID #11) with cost = $1.0/sec
58.962   Sets the submission time to 58.962
58.962   Sets Gridlet status from Created to InExec
58.962   Sets the execution start time to 58.962
80.684   Sets Gridlet status from InExec to Success
80.684   Sets the wall clock time to 21.722 and the actual CPU time to 21.722
80.684   Sets the length's finished so far to 5000.0
```

Listing 7 Example of output statistics for User_9

The basic report-file for carrying out further performance analysis is the GridSim_stat.txt file, in which all the job (Gridlet) submission and receiving events are registered with association to model time. Below is given an example of the form in which these data are presented in the GridSim_stat.txt file:

```
10.760896679999997      "Submit Gridlet_0 to Nyx"       User_7
10.770896679999995      "Submit Gridlet_0 to Cluster"   User_8
10.780896679999996      "Submit Gridlet_0 to Altair"    User_9
112.73105668          "Received Gridlet_0 from Cluster"     User_0   6.0
112.77105668000002    "Received Gridlet_0 from Cluster"     User_4   5.96
113.95648613163279    "Received Gridlet_0 from Masquerade"  User_2   7.098
```

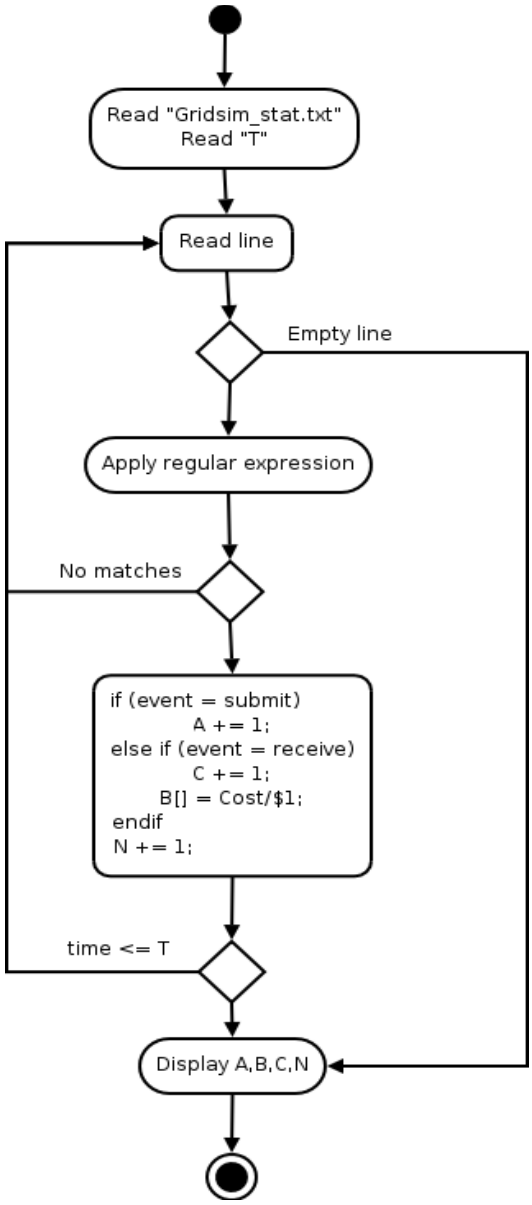

Fig. 6 Simplified diagram of pre-processing logic for basic metrics calculation.

Listing 8 Example of output report data about submission/receiving of jobs by users

In Listing 8 in the first column we see the time in association with the modeling time scale. In the second column we have the description of the event (Submit/Receive), number of the job (0/1, depending on the workload model), the name of the Grid resource (Nyx/Custer/Altair/Masquerade). The third column contains the name of the user, which makes the action. In the fourth column we receive the cost of the execution (for successfully executed jobs) which amounts to the product of actual execution time and the cost of one second of execution time for each specified resource.

Testbed model's statistics analysis. The analysis of the statistic received after running our GridSim testbed model is performed by parsing the statistic files (for each iteration of the experiments scenario) via a script written in programming language PHP. The file containing this script is named parse.php, its full code content that will give you ideas on parsing algorithm is given in Appendix.

The simplified diagram of pre-processing logic implemented in parser.php for basic metrics (in terms of OA) calculation is shown at Fig. 6.

The performance of this script can be described as follows.

— The script receives the model time limit as the input parameter.

— The script reads the input statistics file.

— The parsing takes place for each line, deriving the statistics for each specified host and entity (user).

Below in the Listing 9 an example of the parse-report received after parsing a statistics file is given. In this case we parsed the statistics file stats_200_3 for 200 users, each submitting 3 Gridlets. The model time limit set for parsing is 300.

```
# php -f parse.php stats_200_3 300

Using model time limit: 300
Reading "stats_200_3"...
Found 438 line(s)
Processing...Successfully processed 397 of 397 line(s)
Acquired stats for 5 host(s)
Acquired stats for 220 user(s)

T = 300
N = 297

--- IKD ---
A = 47  B =     C = 0
--- Cluster ---
A = 66  B = 6   C = 52
--- Masquerade ---
A = 61  B = 115.905583383       C = 47
--- Altair ---
A = 52  B =     C = 0
--- Nyx ---
A = 71  B =     C = 0
```

Listing 9 Example of output parse-report

The data received after parsing contains the values of A, B, C, N and T parameters, where N – mean number of requests in the system, T – length of the observation interval, A – number of arrivals during the observation interval, B – length of time the system or resource was observed to be busy during the observation interval, C – number of completions during the observation interval

Analogous parsing was done for statistic files (gridsim_stat.txt) received after iterations of our experiment performed for 10, 20, 30, 40, 60, 80, 100, 150, 160, 180,200, 220 and 250 users, the number of Gridlets submitted by each user is constant and taken as 3 for all these iterations. The model time limit taken for parsing of all these experiments is constant and equals to 300.

Operational analysis. Having received the results testbed model's statistics analysis, including the values of A, B, C, N and T parameters from experiment iterations performed for 10, 20, 30, 40, 60, 80, 100, 150, 160, 180,200, 220 and 250 users, where the number of Gridlets submitted by each user is constant and taken as 3 for all these iterations.

Further operational analysis of the obtained data was made based on the following formulas: Throughput – X=C/T, Utilization – U=B/T, Average service requirement – S=B/C.

The results of operational analysis are presented in the following tables.

OA metrics based on pre-processed data Table 2

N	X	U	S
24	0,067	0,264	3,962
44	0,107	0,356	3,335
65	0,163	0,628	3,845
79	0,190	0,866	4,560
115	0,263	1,129	4,289
133	0,323	1,643	5,083
147	0,243	0,670	2,754
210	0,243	0,221	0,910
232	0,273	0,445	1,627
241	0,337	0,469	1,393
297	0,330	0,406	1,231
330	0,320	0,333	1,042

Conclusions

Nowadays, current trends that lead to the development of distributed systems for complex problems solving with the use of high-performance computing are observed. Grid represents an appropriate technology that enables integration and management of geographically distributed informational and computational resources. But the use of such distributed resources within Grid systems requires the solution of a number of problems considering management of resources and scheduling of applications. In this paper simulation modelling of the Grid system testbed was done in order to investigate and analyze its productivity. For this purpose, GridSim modelling package was used. An approach based on the operational analysis was used for the performance analysis of the system. Such approach enabled qualitative and quantitative estimation of the Grid system performance.

1. I. Foster, C.Kesselman, S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," International Journal of High Performance Computing Applications, 15 (3), pp. 200-222, 2001.
2. W. Gentzsch, "Special issue on metacomputing: From workstation clusters to internet computing," Future Generation Computer Systems, 15, 1999.
3. L. Fusco, P. Goncalves, J. Linford, M. Fulcoli, A. Terracina, G. D'Acunzo, "Putting Earth-Observation on the Grid," ESA Bulletin, 114, pp. 86-91, 2003.
4. Envisat. — http://envisat.esa.int.
5. Japan Aerospace eXploration Agency (JAXA), http://www.jaxa.jp/index_e.html.
6. Information Power Grid (IPG), http://www.ipg.nasa.gov.
7. GridSim. — http://www.buyya.com/GridSim.