

Simulating the Distributed Ontology Framework in the Semantic Grid Environment with GridSim

Andrew Flahive, Wenny Rahayu
La Trobe University, Australia
{A.Flahive, W.Rahayu}
@latrobe.edu.au

Bernady O. Apduhan
Kyushu Sangyo University, Fukuoka, Japan
bob@is.kyusan-u.ac.jp

David Taniar
Monash University, Australia
David.Taniar@infotech.monash.edu.au

Abstract

There are many simulation environments around that can be used to simulate components of Grid Computing. There are also a few simulation environments that allow for data intensive jobs to be simulated. GridSim, with the Data.Grid extension, combines both of these simulation environments to allow simulation of data intensive jobs using Grid resources. GridSim is explored in this paper with an aim to simulate the Distributed Ontology Framework (DOF) in the Semantic Grid Environment. The DOF requires many data resources to be located as well as large scale processing to take place on the collected data. Several simulation environments are discussed and the similarities between the DOF and the GridSim environment are explored. Several simulations are carried out to show whether or not GridSIM is a suitable simulation environment for simulating the Distributed Ontology Framework.

1. Introduction

The large-scale problem solving mechanism in Virtual Organizations is commonly known as the Grid[15]. Large computational and storage devices are linked globally to form a network of powerful resources. Grid Computing focuses on the amount of computations that can be salvaged across a large distribution of heterogeneous machines. A Data Grid is similar to a Grid but enables large data transfers between the heterogeneous machines[14, 3, 13].

The Semantic Grid is an extension of the current Grid in which information services are given well defined meaning[16]. It is a combination of high semantics and large processing power. The Semantic Grid allows people and computers to work together to solve large scale

knowledge intensive tasks. The use of semantics allow the computers to understand the information they contain.

Both the Semantic Grid (SG) and the Data Grid (DG) deal with large amounts of data transfers between resources. They both aim to support e-scientists and engineers from different universities and research laboratories to collaborate with one another to solve large-scale computational and data intensive problems. The main difference between the two is that the focus of the SG is on structured knowledge sharing through semantics whereas the DG has a wider focus, encompassing any Grid types that involve transferring large data sets between networked entities[22]. However, in simulation terms, the Data Grid and the Semantic Grid are the same, the only difference is the information that is stored and the methods used to store it.

Given the size of the data sets, it is impossible for a human to make decisions about where the data needs to be at any particular moment. Due to the high latency of the internet, many models involving the Semantic Grid (or any Grid) cannot accurately be designed, as large data transfers are required. Simulation allows us to set up an environment where various models can be experimented with and configured in a controlled and repeatable environment. Simulation often returns a quicker result than if the experiment were actually carried out.

The Distributed Ontology Framework (DOF) is a model designed to help e-scientists and engineers tailor large ontologies in the Semantic Grid Environment[12]. Being able to tailor these large ontologies in the Semantic Grid enables knowledge to be extracted faster, cheaper and more efficiently. However, such a system cannot be easily configured to allow for efficient results in any situation. This paper describes the simulation of the DOF in the Semantic Grid Environment using the GridSim package.

2. Background

2.1. Earlier Work on the DOF

E-Scientist and Engineers require methods to tailor large base ontologies to extract a smaller sub-ontology that better represents their domain of interest. Often ontologies are too large and cumbersome to use in their entirety and so having the option and ability to extract certain information is very valuable. Often in large base ontologies, new knowledge is sought by applying certain algorithms. Analyzing the semantic data by hand would take forever and the knowledge that is hidden within would never be found. Being able to tailor large ontologies opens up a more efficient world where knowledge is sought and discovered at a faster rate leading to new findings that advance science and technology. However, ontology tailoring is not an easy task and the processing requirements are huge. No current single computer could efficiently tailor a large ontology.

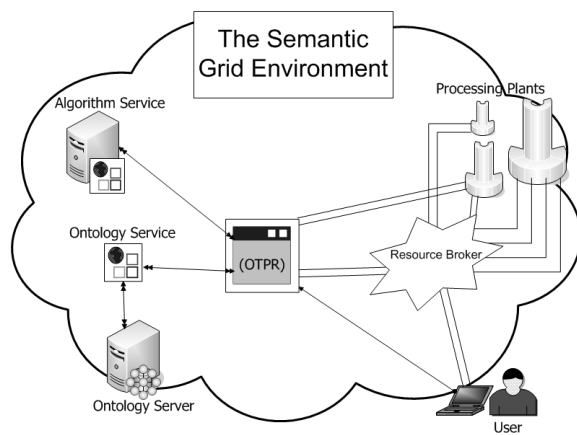


Figure 1. The Distributed Ontology Framework

Our previous work on the DOF was aimed at assisting e-scientists and engineers to tailor their ontologies using resources available in the Semantic Grid Environment. The framework itself is made up of five categories to assist in the tailoring of large base ontologies with a goal of producing smaller tailored ontologies that adhere to a set of initial criteria. The Five main categories are; (i) Ontology Processing, (ii) Ontology Location, (iii) Ontology Connection, (iv) User Connection and (v) Algorithm Location. These categories assist to describe the framework by dividing the main issues into distinct groups that are much easier to discuss.

A smaller tailored ontology is a subset of the larger base ontology and is itself a complete and valid ontology. Many algorithms may be used to tailor the large base on-

tology. The Ontology Tailoring Program Resource (OTPR) is a software resource that has the ability to use the algorithms to tailor the large ontologies. These algorithms may be simple or complex. Depending on the algorithm complexity and the size of the base ontology the time taken to tailor an output could be as short as a few seconds or take several months. The framework uses resources available in the Semantic Grid environment to obtain such results efficiently and at low cost to the user.

An example of the DOF is shown in Figure 1. A more detailed introduction to the Distributed Ontology Framework can be found in [12, 11].

2.2. GridSim and the Data.Grid Extension

GridSim has been a popular simulation environment for simulating workloads in the Grid environment. GridSim was originally conceived by Buyya[5] and is now used all over the world by many institutions to simulate specific Grid scenarios. GridSim has been used to simulate many projects dealing with Grid and cluster computing.

The GridSim toolkit allows modeling and simulation of entities in parallel and distributed computing (PDC) systems-users, applications, resources, and resource brokers (schedulers) for design and evaluation of scheduling algorithms[6]. Grid loads are used to simulate the computational size of particular jobs that are required to be completed. GridSim can use space and time shared methods in its computational calculations where specified. These jobs can be processed in a controlled environment where a grid of heterogeneous resources are available over any type of connection.

GridSim was recently extended to include simulation large data transfers and processing. The DOF is able to be simulated in ways that the original GridSim could not. The Data.Grid extension to GridSim allows for large and small files to be replicated across the grid environment. It allows the processing loads of jobs that require files to be transferred before the processing begins. It is able to calculate the computations to process the required loads across a Data Grid environment and include file transfer and replication costs. The Data.Grid extension of GridSim (DataGridSim) allows integrated studies of on-demand replication strategies with job scheduling on available resources[2]. The extension allows us to increase the number of situations that can be simulated by allowing multiple replications of the same ontology to exist in the Grid Environment. Without the extension, many situations could not be simulated accurately.

3. Related Simulation Environments

Although the new GridSim package seems to be the complete simulation solution, several other simulation packages were looked at to determine their ability to simulate the DOF. This section discusses the other simulation packages and their ability to appropriately simulate the DOF in the Semantic Grid Environment.

Having the ability to simulate a Grid environment is the first key requirement of the DOF. Being able to configure a network topology of resources and computational power is very important. BeoSim[18] is designed to model a mini-grid and has a fully configurable cluster/grid topology. SimGrid[7] and Bricks[24] also have configurable grid topologies and are used for the simulation of distributed and parallel application scheduling. MicroGrid[23] again has fully configurable grid topology but is aimed more toward the simulation of Grid middleware applications and network services. None of these simulation packages, however, are able to simulate the transfer of large data elements through the Grid network. As large ontologies are required to be transferred in the DOF and many other data sources are required to be used, these simulators are less than ideal to simulate the complex data flows of the DOF.

The ability to simulate data transfers is another important feature of the DOF that requires simulation. Monarc[19] is designed to provide a realistic simulation of distributed computing systems, customized for specific physics data processing requirements. ChicSim[1], like Monarc is designed to provide a wider variety of scheduling and replication algorithms in a Data Grid type environment. ChicSim manages job requirements of certain data-files to be present before certain data processing jobs can begin. Likewise, Gridnet [17] manages the replication runtime components and evaluates the data access gains, creation and maintenance costs of replicas before moving or copying any data. Although these simulation packages are able to simulate the data transfer needs of the DOF in a more suited Data Grid environment, they all lack the ability to manage more computational loads between the computational resources available in the environment.

OptorSim[4] is probably the next best simulator available to simulate the DOF. It is able to simulate a Data Grid environment and is also able to manage files and process loads based on data transfers. Like GridSim, it is based on the EU DataGrid Project[9, 20] and provides a very realistic data grid environment. It's goal is to evaluate the impact that different algorithms have on the throughput of grid workloads. The DOF however, requires computation loads as well as data transfer loads to be computed. GridSim is able to cover the same number of features required by the DOF as OptorSim, but enables a greater variety of computational workloads to be simulated as well. The ability to

combine computational workload management as well as data transfer management and replication put GridSim in a far better position to simulate the DOF more completely than any other currently available simulation package.

4. Proposed Evaluation of the DOF

In order to evaluate the Distributed Ontology Framework in a controlled and configurable environment, simulation was chosen as the best method. Previous sections have discussed various simulation packages that may be used to model various data intensive and computationally workloads in different grid environments. From these discussions GridSim was chosen as the simulation package that could best describe the DOF and prove it's flexibility.

Earlier papers by the authors[10, 11, 12] have introduced the proposed DOF and have discussed in detail the issues surrounding it in real world situations. This paper extends this work by presenting a simulation of the DOF in the Semantic Grid Environment. This section of the paper develops the simulation environment and makes connections between the components of the DOF and the GridSim package. A time line is also provided to show the sequence of events that occur during a typical simulation run.

4.1. Simulation Environment

The implementation environment was Java 1.4.2 (J2SE)[21] running on WinXP with 512MB RAM. GridSim 4beta[8] was used, as this version has the Data.Grid extension[2].

4.2. DOF to GridSim Mapping

Unlike many of the other simulation packages investigated, DataGridSim has many features that aid the simulation of the Distributed Ontology Framework. This section goes through some of the main features in GridSim that are used to simulate the Distributed Ontology Framework in the Semantic Grid Environment.

One of the main categories in the DOF is the location of the ontology[12]. This is where an ontology is initially located with either the User or with a remote resource. GridSim allows network of resources to be created where large data files can be added to any of them to represent an ontology. The network connection between the ontology and the main processing unit(s) can be adjusted to simulate the different situations described in [10].

Another main function of the Distributed Ontology Framework is the type of connection made to the ontology. This is where the ontology is either transferred as a whole to the processing unit or accessed through a remote set of commands. In the latter case, GridSim allows

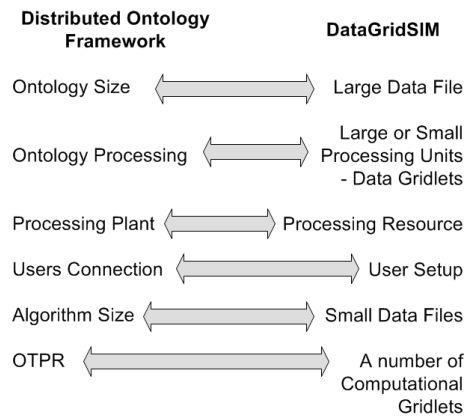


Figure 2. Mapping the DOF to GridSim

us to store many files instead on one large file to represent the ontology. DataGridlets are used as the workload to simulate the ontology tailoring and any number of files are required by the DataGridlet before it can start to be simulated. Therefore, by combining different network scenarios with various numbers of DataGridlets the Ontology location and Ontology connection categories can be accurately simulated.

The algorithms in the DOF can be seen as small files that could be located anywhere in the Semantic Grid Environment. GridSim allows us to spread the algorithms across the Semantic Grid with various connection speeds and connection types. Again by creating various DataGridlets the algorithm files are collected before being used to simulate each individual algorithm. The DataGridlets also determine how much tailoring is required based on the files collected and the size of the ontology.

Two categories in the DOF allow for the use of a Resource Broker. GridSim has implemented a Replication Manager (RM) to control the whereabouts of certain resources. The RM does not take over the role of the Resource Broker or the OTPR, it simply manages their involvement and produces an output relating to their various involvements. The RM helps by adding/deleting an ontology file or replicator and sends files between different entities like the User, OTPR and Processing Plant.

An added bonus that DataGridSim allows the DOF to simulate is the improvement in processing speed if the ontology were located at multiple sites instead of just one. The Replicator Catalog helps locate multiple instances of resources that store a particular piece of data. The catalog may also know the location of various OTPRs that match certain ontology tailoring abilities. By adjusting these variables in a controlled and repeatable environment, GridSim allows us to demonstrate the benefits of the DOF given various environment constraints.

4.3. Time-line of Events

DataGridSIM has a certain order of operations where certain things must occur in a certain order. The DOF also has an order of operations where certain things must occur before other entities can do their job. It is important to match these two orders of operations so that an accurate simulation can occur. Using the mapping discussed in the previous section, a time-line of events is presented to show how the simulations are carried out as well as showing what happens at each stage of the simulation.

1. The User feature in DataGridSim allows the User to compile a list of jobs to be completed and files to retrieve before any tailoring can be completed. It can also specify which order the jobs should be completed and the individual files required. The User sets these up as Gridlets and DataGridlets before requesting help from the Resource Broker to find the best OTPR available.
2. The Replication Manager (RM) receives a list of items required for the job. These things include: location of algorithms, location of ontology, user requirements to be met etc.
3. The Replica Catalog (RC) checks the location of all of the files required. It is also able to check if the files required are located anywhere else in the Semantic Grid Environment.
4. The RC compiles a list of these locations.
5. The RM is able to then make a decision of the processing of the ontology as there may be one location that combines all of these data files.
6. If all of the data files are found at the one location (including the OTPR) then the RM suggests processing the task at this location. Otherwise it collects all of the data required and sends it to the OTPR that is least in demand and has access to suitable processing facility.
7. All of the data transfers are monitored in the simulation and added to the processing time taken to tailor the different sized ontologies using different and varied algorithms. The different sized algorithms are represented by different sized files and so too is the size of the ontology. The amount of tailoring required is a combination of the size of the ontology file and the various sizes of the algorithm file(s).
8. If a processing plant is required, the time taken to find one, transfer the appropriate data and process the workload is taken into consideration and added to the overall time.
9. The new tailored ontology is then sent back as a file to the user.

5. The DOF Simulation

This simulation attempts to show how the replication of the base ontology can lead to improved tailoring times. This is due to the fact that there is a greater chance that the ontology can be copied up to the OTPR quicker so that it can be tailored faster. This is a good simulation to show as it involves all of the components of the framework.

There are two parts to this simulation, Experiments 1 and 2. Experiment 1 simulates a situation where the ontology is copied up from the User to the OTPR. Experiment 2 simulates a situation where the Ontology is copied up to the OTPR from a remote resource that has a much faster connection than the User. This change should result in a more efficient ontology tailoring experience.

5.1. Experiment Setup

This section describes how the experiment was configured and how each of the components of the DOF were created and made ready for simulation. To make sure the simulation covers the whole DOF, the five categories in the DOF are used to describe the setup features. The overall Semantic Grid Environment is then described in terms of a network topology.

5.1.1. Ontology Processing A Processing Plant is required as there is a large amount of processing to complete. There are two main processing plant resources available in this simulation; Processing Plant 1 and Processing Plant 2. These are shown in the top and top right corner of Figure 3. These two resources are used in both experiments.

To simulate the initial setup cost of tailoring of the Ontology, one large DataGridlet was created and submitted. This DataGridlet required the collection of all of the algorithms and the ontology before it could begin to be processed. The size of the ontology (80GB), directly related to the size of this DataGridlet. If the size of the ontology was increased so too was the size of this Gridlet.

For the other ontology tailoring computations required, a number of computational Gridlets were set up and used in combination with the DataGridlets. This provides for a more accurate tailoring simulation of the OTPR.

5.1.2. Ontology Location The Ontology is initially located at User location only. Experiment 1 includes the copying up of the ontology from the User location. In Experiment 2, the Ontology is initially located at a remote resource and is copied up from this location. This location of the ontology has a much faster connection to the OTPR and is expected to reduce the overall time taken to complete this experiment. The change in location of the Ontology is the only difference between Experiment 1 and Experiment 2.

5.1.3. Ontology Connection The Ontology is copied up to the OTPR's local work space as a whole. The simulation was designed so that the ontology could fit into the local storage space of the OTPR.

5.1.4. Users Connection The User was created with a slow connection to show that it is less efficient to house the ontology with the User location in this situation. It also is meant to show that an efficient Ontology Tailoring experience can be had by the User, regardless of their connection speed, as long as there is a fast connection to the ontology.

5.1.5. Algorithm Locations Algorithms are located at the User (Algorithm Location A), 2 remote locations (Algorithm Location B and C) and at the OTPR (Algorithm location D). A DataGridlet was added for each of the algorithms to simulate the effect of the algorithms on the overall tailoring process. Each DataGridlet requires one of the Algorithms to be located and stored locally before it can begin to be processed. This simulates algorithms that are distributed around the Semantic Grid Environment.

5.2. Overall Network Topology

Figure 3 shows an overview of the Semantic Grid environment that was set up for this simulation. Experiment 1 has the Ontology location B component excluded during simulation. Experiment 2 includes this extra resource.

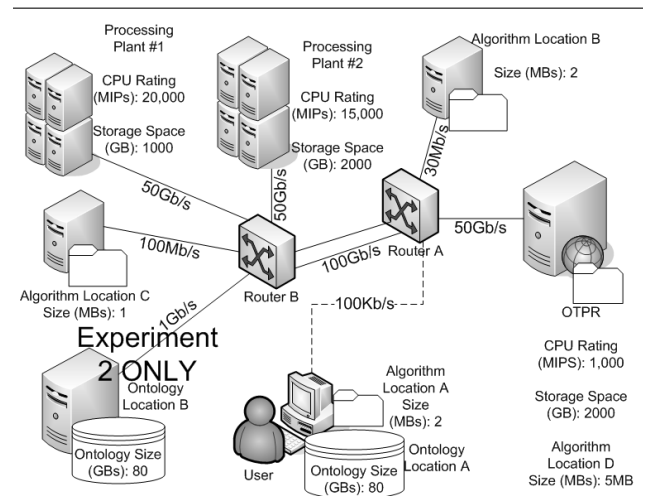


Figure 3. Network Topology of Simulation

In total, 7 entities were created but, only the two processing plants were sent jobs to complete as these are the largest and most efficient processing resources. The other resources just provide files like the ontology or algorithms. The OTPR is the resource that controls the processing of the jobs of the processing plants.

5.3. Experiment 1

As a great deal of processing is required the ontology is required to be copied up to the OTPRs local storage space before processing takes place. This experiment shows the time taken to process the ontology as the ontology size increases.

In this simulation the Ontology is located at the Users location only. The User has a very slow connection to the Semantic Grid in comparison with the other resources. In the actual simulation, all of the network speeds were lowered by a factor of 100 and the ontology was reduced by a factor of 100 also. Changing these figures meant that the simulation stayed within the allowed ranges by DataGridSIM and still allowed a comparable simulation.

As can be seen from the top line of the graph in Figure 4, for a large ontology of 80GB the ontology tailoring process is slowed dramatically compared with the smaller ontologies. As mentioned this is due to the fact that the ontology must be copied up from the Users location to the OTPR before processing can begin.

5.4. Experiment 2

In this next simulation, the Ontology is now located at a remote location as well as at the User site. This new location of the ontology has a fast connection to the OTPR and is able to be transferred quicker. The OTPR realizes that it is more efficient to retrieve the ontology from this new location than from the User.

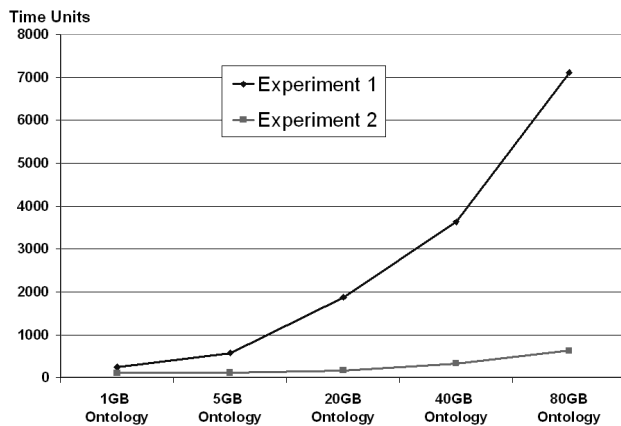


Figure 4. Results of Simulation

The bottom line of the graph in Figure 4 shows the results from Experiment 2. The efficiency is improved in this simulation because the ontology tailoring isn't held up by the copying up of the ontology from the User.

6. Discussion

The fact that the same ontology could exist at several remote locations was not something that was originally considered when the DOF was developed. For instance, if it is too hard to move the ontology to the OTPR why not move the OTPR to the Ontology, as long as there is an appropriate processing facility available at the same location. At the very least, DataGridSIM provides the option to use a closer version of the same ontology to allow a speedier tailoring experience.

Simulating poor networks is an important simulation in that the Semantic Grid is a dynamic environment where network speeds change minute to minute and components appear and disappear in seconds. The DataGridSIM package was able to simulate bad connections to the user and other components through its network topology component. It was able to distinguish when a connection to one resource was better to use than another and alter the processing as required. It allowed accurate simulations of Ontology workloads that are both computationally intensive as well as data intensive. The ability to gather many files before a processing element can begin is a huge advantage when attempting to simulate the Distributed Ontology Framework in the Semantic Grid Environment

Simulating the various ontology sizes was an important simulation as we needed to see if the framework could handle large and small ontologies. The GridSIM package is also able to simulate user imposed time limits and budgets. These are also important factors when looking to simulate the DOF. The contributions made in this paper will allow us to extend the number simulations so that a complete exploration of the DOF can occur.

7. Conclusion

Previous papers by the authors have set up the DOF to enable large base ontologies to be tailored in the Semantic Grid Environment. This paper established a simulation environment that enabled the Distributed Ontology Framework to be simulated with reasonable accuracy. The simulation package used was the new GridSIM package that includes the Data.Grid extension. Several experiments were carried out that proved that many aspects of the DOF could be modeled and simulated using this package.

The simulations proved that by having more than one ontology in the Semantic Grid Environment, the chances of obtaining an efficient result increased regardless of the size of the ontology. The experiments dealt with poor networks and how the DOF coped given slow connections between some of the components. The DOF was designed to be flexible so that it allowed for many situations to be modeled. The GridSIM package with the Data.Grid extension allowed us

to simulate this flexibility using the one simulation package. Without this extension it would be hard to gauge the benefits if two simulation packages had to be used, one to simulate various loads in the grid scenario and the other to simulate data transfers etc.

Now that a solid simulation environment has been established, further simulations can take place to test the validity of a Service Oriented Approach to the Distributed Ontology Framework. DataGridSIM could not help simulate a possible implementation of the DOF using this approach at this time. The easiest way to test such a method would be to implement and deploy it. Extending the DataGridSIM simulations to include real Web Services as data resources would help to test approach whilst giving added realism to the simulations.

References

- [1] Chicsim: The chicago grid simulator. <http://people.cs.uchicago.edu/~krangana/ChicSim.html>, 2006.
- [2] A. Sulistio, U. Cibej, B. Robic and R. Buyya. A tool for modelling and simulation of data grids with integration of data storage, replication and analysis. Technical report, Grid Computing and Distributed Systems Laboratory, University of Melbourne, Australia, November 2005.
- [3] B. Allcock, J. Bester, J. Bresnahan, A. L. Chervenak, I. T. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnel, and S. Tuecke. Secure, efficient data transport and replica management for high-performance data-intensive computing. *CoRR*, cs.DC/0103022, 2001.
- [4] W. Bell, D. Cameron, L. Capozza, P. Millar, K. Stockinger, and F. Zini. Optorsim: A Grid simulator for studying dynamic data replication strategies. *International Journal of High Performance Computing Applications (IJHPCA)*, 17(4):403–416, 2003.
- [5] R. Buyya and M. Murshed. Gridsim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing. *Concurrency and Computation: Practice and Experience (CCPE)*, 45(14):13–15, 2002.
- [6] R. Buyya and A. Sulistio. Gridsim: A grid simulation toolkit for resource modelling and application scheduling for parallel and distributed computing, 2006. <http://www.gridbus.org/gridsim/>.
- [7] H. Casanova. Simgrid: A toolkit for the simulation of application scheduling. In *CCGRID '01: Proceedings of the 1st International Symposium on Cluster Computing and the Grid*, page 430, Washington, DC, USA, 2001. IEEE Computer Society.
- [8] U. Cibej and A. Sulistio. Gridsim.datagrid: A data grid extension for gridsim toolkit. <http://www.gridbus.org/gridsim/datagrid/index.html>, 2006.
- [9] European Centre for Nuclear Research. The datagrid project. <http://eu-datagrid.web.cern.ch/eu-datagrid/>, 2004.
- [10] Flahive, A., Apduhan, B., Rahayu, W. and Taniar, D. The distributed ontology framework: Case studies in the semantic grid environment. *International Conference on Parallel and Distributed Processing Techniques and Applications*, pages 231–237, 2005.
- [11] Flahive, A., Apduhan, B., Rahayu, W., and Taniar, D. A distributed ontology framework in the semantic grid environment. *Proc. AINA-2005/International Workshop on Information Networking and Applications (INA)*, pages 193–196, 2005.
- [12] Flahive, A., Rahayu, W., Taniar, D. and Apduhan, B. A distributed ontology framework for the grid. In K.-M. Liew et al, editor, *The 5th Int'l Conf. on Parallel and Distributed Computing, Applications and Technologies (PDCAT)*, volume LNCS 3320, pages 68–71, Singapore, 2004. Springer-Verlag Berlin Heidelberg.
- [13] I. Foster. The grid: A new infrastructure for 21st century science. <http://www.aip.org/pt/vol-55/iss2/p42.html>, 2002.
- [14] I. Foster and C. Kesselman. *The Grid : Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers, 1999.
- [15] I. T. Foster. The anatomy of the grid: Enabling scalable virtual organizations. In *Proceedings of the 7th International Euro-Par Conference Manchester on Parallel Processing*, pages 1–4. Springer-Verlag, 2001.
- [16] C. Goble and D. D. Roure. The grid: An application of the semantic web. *CACM*, 31(4), December 2002.
- [17] B. S. Houada Lamahamedi, Zujun Shentu and E. Deelman. Simulation of dynamic data replication strategies in data grids. In *WSC '00: Proceedings of the 32nd conference on Winter simulation*, pages 1794–1801, San Diego, CA, USA, 2000. Society for Computer Simulation International.
- [18] W. M. Jones, L. W. Pang, D. C. S. Jr., and W. B. L. III. Job communication characterization and its impact on meta-scheduling co-allocated jobs in a mini-grid. In *18th International Parallel and Distributed Processing Symposium (IPDPS 2004)*, 2004.
- [19] I. C. Legrand and H. B. Newman. The monarc toolset for simulating large network-distributed processing systems. In *WSC '00: Proceedings of the 32nd conference on Winter simulation*, pages 1794–1801, San Diego, CA, USA, 2000. Society for Computer Simulation International.
- [20] B. Lesyng, P. Baia, and D. Erwin. Eurogrid: European computational grid testbed. *J. Parallel Distrib. Comput.*, 63(5):590–596, 2003.
- [21] S. Microsystems. Java 2 platform, standard edition. <http://java.sun.com/j2se/1.4.2/download.html>, 2006.
- [22] S. Cochrane, Et. al. Understanding data-grid architecture for higher education. Technical white paper, Sun Microsystems Inc, February 2005.
- [23] H. J. Song, X. Liu, D. Jakobsen, R. Bhagwan, X. Zhang, K. Taura, and A. Chien. The microgrid: a scientific tool for modeling computational grids. In *Supercomputing '00: Proceedings of the 2000 ACM/IEEE conference on Supercomputing*, page 53, Washington, DC, USA, 2000. IEEE Computer Society.
- [24] Takefusa, A., Casanova, H., Matsuoka, S. and Berman, F. A study of deadline scheduling for client-server systems on the computational grid. In *Proceedings of the 10th International Symposium on High Performance Distributed Computing (HPDC-10)*, pages 406–415. IEEE Computer Society, 2001.