# Gaining insight in the analysis of performance for Resource Monitoring and Discovery in Grids

D. González Márquez, D. Fernández Slezak, P. Turjanski, and E. Mocskos

Laboratorio de Sistemas Complejos, Departamento de Computación,
Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires
Buenos Aires (C1428EGA), Argentina.

**Abstract.** Nowadays, many High Performance Computing (HPC) centers with thousands of cores are available. The interconnection and synchronization between them is one of objectives of Grid Computing, as all these centers tend to form a global shared computing facility. In such scenario, dynamics of resource information is increasingly complex, and may become almost impossible to be captured using a static hierarchy. One of the key features needed in Grid Computing is to maintain up-to-date information about resources and services. Two important parameters that must be selected are the information refresh and expiration times. In this work, `R` is presented for studying the relation between these important parameters in three policies: Hierarchical, Super-peer and Random. The results reveal an interesting relation between expiration and refresh times in all studied policies. The study of this new parameter is essential to provide a deeper knowledge in new information distribution policies and in the configuration or deployment of existing ones.

## 1 Introduction

With the advances in microprocessors, storage capacity, networking speed and telecommunications in general the world has witnessed an unprecedented progress in information technology, enabling to satisfy the increasing demands of computing power required by scientific and commercial applications.

Many High Performance Computing (HPC) centers are being built and Grid Computing arises as the new computing paradigm[7, 3, 8, 10] for interconnection and synchronized interaction between them, providing the infrastructure required for sharing diverse set of resources, including desktops, computational clusters, supercomputers, storage, data, sensors, applications, and on-line scientific instruments[19].

In this technology, Globus Toolkit 4 (GT4)[9, 1] is one of the most used tool for creating grid environments over different operating systems and computer architectures; which consists of a set of services and applications for managing the different grid features following a layered design.

One of the key features needed in Grid Computing is to maintain up-to-date information related to resources and services in an heterogeneous and highly dynamic environment[4]. The component responsible for obtaining, distributing,

indexing and archiving information about the configuration and state of services and resources is called the Grid Resource Information Service (GRIS). Standard centralized organization approach for the GRIS has several drawbacks[19]:

- Highly prone to a single point of failure.
- Lacks scalability.
- High network communication cost in the links leading to the information server (i.e. network bottleneck, congestion).
- Centralized servers may become bottlenecks or points of failure.

Monitoring and Discovery System (MDS)[2, 6], the specialized services of GT4 for resources monitoring and discovery, uses this kind of centralized approach as the default configuration.

Information technology trends suggest that scenarios will count with a enormous number of shared resources in the Grid, converting the whole planet in a shared computing facility. In such a system, the dynamics of the resource information is almost impossible to be captured using a static manually maintained hierarchical structure[20], which has similar problems to the centralized approach, such as the point of failure, and poor scaling for a large number of users/providers[17, 18].

Therefore, it is necessary to design new policies for discovery and propagation of resource and monitoring information. There is a growing interest in the interaction of the Grid Computing and the Peer to Peer (P2P) paradigm: both work within a very dynamic and heterogeneous environment, where the role and availability of resources may quickly change; both create a virtual working environment by collecting the resources available from a series of distributed, individual entities[18]. Extensive work has been devoted recently to obtain a new standard to replace the hierarchical policy, starting with a proposal for decentralizing grid information services that was made by Iamnitchi et al.[12, 13] where they proposed a P2P based approach for organizing the MDS directories in a flat dynamic P2P network.

Following this trend, a practical approach towards scalable solutions is offered by P2P models[15, 18, 20]. In this sense, many research efforts have been aimed to design new GRIS policies; their theoretical benefits are usually shown using ad-hoc simulation tools to compare to other previously published policies[16].

In a previous work[11], GridMatrix2 was presented as a very powerful simulation tool designed to focus on the development and testing of discovery and monitoring information policies with a friendly graphical user interface and easy-to-use mechanism, based on SimGrid. Using this tool, three different information propagation policies were implemented: Hierarchical, Super-peer and Random. These policies were studied in two different network environments, Clique and Ring topologies, that were automatically built using GridMatrix2. Three metrics that capture and summarize the information propagation behavior were presented: Local Information Rate (LIR), Global Information Rate (GIR) and Global Information Variability (GIV), allowing to get very useful information and insight into the details of the modeled systems.

In this work, R is presented for studying the impact of essential configuration parameters in three policies: Hierarchical, Super-peer and Random. The final objective is to have a system that can keep the information as new as possible using the minimum resources.

The rest of the paper is organized as follows: section 2 includes a brief review of metrics, simulation tool and a new metric to gain insight in the dynamic behavior of the systems. The section 3 shows the usage of this new metric, while in section 4 we draw some final remarks.

## 2 Methodology

GridMatrix2[11] offers an easy-to-use graphical environment for simulate information propagation policies, it is based on SimGrid[5], a framework that provides the core for the simulation of distributed applications in heterogeneous distributed environments. GridMatrix2 interacts with SimGrid, and provides a full-featured graphical interface that helps configure the network topologies and behavior. It includes an interface for scripting the node behavior in *Python*, giving access to a very powerful and complete environment to add new functionalities.

To simplify the analysis and implementation of the simulations, the messages obtained by each node contain no details about the *real* resource information, only the expiration time (time-to-live) and the amount of carried information.

The message between each host has two parts: one with fixed size (similar to the header of a real network message) and one in which the size is proportional to the resource information being transmitted (constant size for each resource). In this way, we have no need to introduce the resource details, focusing only on the very rich dynamics of the system. The size of the messages was selected based on the widely used monitoring tool `Ganglia`[14].

### 2.1 Measuring the System Behavior

The system dynamics can not be capture without considering the time as an unavoidable part of the information value. Using the resource information for scheduling purposes forces to take into account the age of the information. It is obvious that a task should be scheduled to the *appropriate* node using the most recent information. In this context, in order to compare the different policies, we define the following dimensionless information rates that capture the change in the total information, the amount of grid information available in each node and the age of this information, and the variability of this information among the nodes:

- LIR: this coefficient is bounded between 0 and 1 and captures the amount of information that a particular host has from all the entire grid in a single moment, weighting the expiration time. A value of 1 would mean the host

knows every single piece of information about the whole grid and is as new as it can be. The definition of LIR for the host $k$ is:

$$LIR_k = \frac{\sum_{h=1}^{N} \frac{(expiration_h - age_h)}{expiration_h} \cdot resourceCount_h}{totalResourceCount}$$

where $N$ is number of hosts in the system, $expiration_h$ is the expiration time of the resources of host $h$ in host $k$, $age_h$ is the time passed since the information was obtained from that host, $resourceCount_h$ is the amount of resources in host $h$ and $totalResourceCount$ is the total amount of resources in the whole grid. The expression $expiration_h - age_h$ is similar to the well known parameter *time-to-live* from the `IP` protocol and, due to its definition, can only has no negative values.

- GIR: this coefficient is also a value between 0 and 1, and captures the amount of information that the whole grid knows of itself. It is calculated taking the mean value of every node's LIR.
- GIV: this coefficient is the standard deviation of GIR. It measures the variability of GIR in the system (less is better).

### 2.2 Information Propagation Policies

Based on the work of Mastroiani et al.[16] three paradigmatic policies were selected to be evaluated with GridMatrix2.

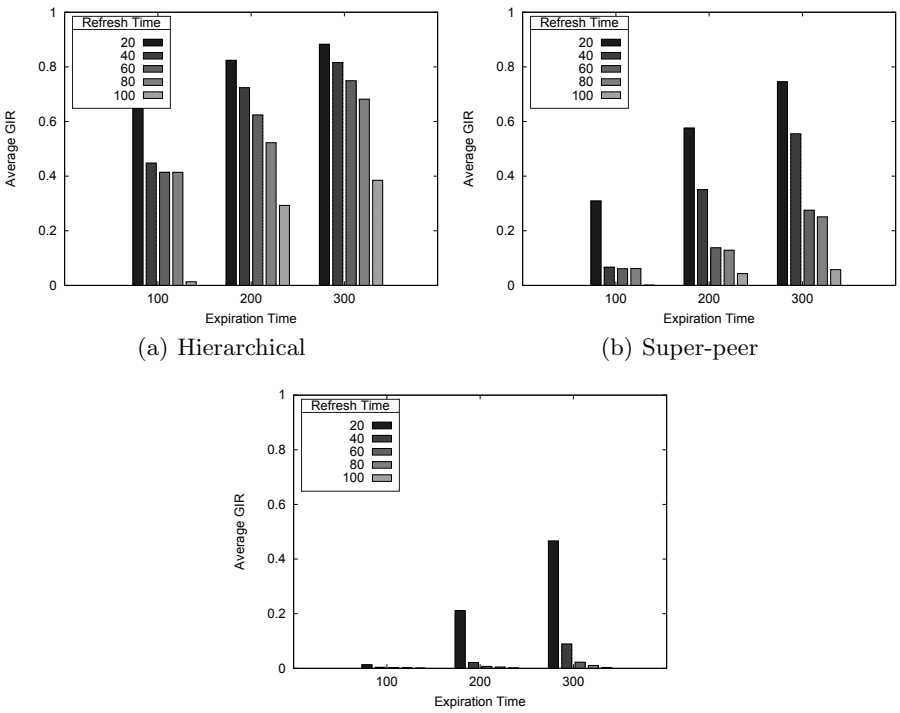
- **Hierarchical:** in this kind of policy, a hierarchy is established beforehand and the resource information is sent using this fixed structure. In this way, the nodes in the top of the hierarchy exchange information with the ones below them. As mentioned above, MDS actually uses this type of information propagation policy[2].
- **Random:** there is no structure at all, every node chooses randomly a neighbor to get the information from. This policy does not save information about quality of each query and the distribution of queries is uniform. All the nodes can query every other connected node[12].
- **Super-Peer:** A super-peer network operates like an unstructured P2P, but some nodes are defined as *super-peers* working like servers for a subset of nodes and as peers in the network of super-peers. In this way, a two level structure is defined such that the *normal* nodes are allowed to *talk* only with a single super-peer and the cluster defined by it. An unstructured P2P is a degenerated super-peer network, where the cluster size is exactly one[21]. In this work, hosts send information to a local super-peer and this information is distributed among the others using Random policy.

### 2.3 A New Metric

The main objective of a resource distribution information policy is to carry the information as fast as possible to all the nodes composing the system using the least amount of resources (i.e. network bandwidth).

As was mentioned before, the resource information has an associated expiration time: knowing that there was free memory or idle CPU in certain node 20 hours ago result in useless information and in a possible poor scheduling strategy. Setting up a distribution policy involves determining the way in which a node *talks* to a partner and how often they exchange information (i.e. information refresh time). These two parameters must be studied in any given configuration to obtain conclusions about the behavior of the selected policy and to determine the optimal values for these parameters.

The figures 1(a) to 1(c) show the mean GIR using a fixed value of information refresh time. The network used in this example has 1000 nodes, corresponding to a exponential-like generated network with the exponent $\alpha = 0.2$ (similar to Internet). The first 100s of simulation are discarded as they correspond to the system setup period.




(a) Hierarchical

(b) Super-peer

(c) Random

**Fig. 1.** Mean GIR in the three studied policies. The refresh time is fixed (showed in the left-top part of each figure), while the information expiration time is swept. The length of each bar represents the mean GIR value, discarding the first 100s as it corresponds to the setup time. The policies show an improvement in GIR when the refresh time is decreased (refresh frequency increased), but the Hierarchical policy presents the best performance in this case.

The figure 1 reveals the relation between expiration and refresh times in all studied policies. If the information received is near to expiration, the overall behavior of the system is poor showing low values of GIR. On the other hand, when the nodes tend to connect often (decreasing the refresh time) the obtained information has a longer valid period, the system performance improves and the resulting GIR values are higher.

In spite of that he general trend is similar in all the studied policies, the obtained mean GIR values are different in each one: Hierarchical shows the best performance in the considered range of refresh time. Super-peer follows it, but its mean GIR notably decreases and presents a more critical dependency with the refresh time. Finally, Random rapidly losses performance with the increase in refresh time.

This imposes the question about the impact and robustness of a policy in the context of the relation between the expiration and refresh times.

To aim this issue, a new metric is introduced: `R` is defined as the relation between expiration and refresh time, i.e. how often the nodes communicate and exchange resource information. It acts as a normalization method used to abstract the duration of the exchanged information in terms of refresh cycles:

$$R = \frac{expiration\_time}{refresh\_time}$$

According to the definition, when `R` is close to 1, the expiration and refresh time will be very similar, leading to a system in which the exchanged information quickly expires rendering it almost useless. In this extreme case, the nodes would count only with the information about its own resources and, depending on how close to 1 is `R` this would be a situation in which all nodes would have only information from their own resources or, at most, with its direct neighbors.

The opposite limit case is identified with very high `R` values: this scenario represents that the exchanged information has a very long time to live allowing it to reach very distant nodes and resulting in very good system performance.

## 3 Results

In this section, we show the utility of the presented metric. The used scenario consists in an exponential generated network ($\alpha = 0.2$) with 1000 nodes. The simulation is executed for 1000s. Three policies are studied: Hierarchical, Random and Super-peer.

### 3.1 Using `R` to study different policies

Figure 2 shows the global system behavior of the considered policies. Each point in the graph represents the mean GIR value of a complete simulation.

A low `R` value means that the resource information flowing has a small associated time to live, rapidly becoming useless. The three considered policies show

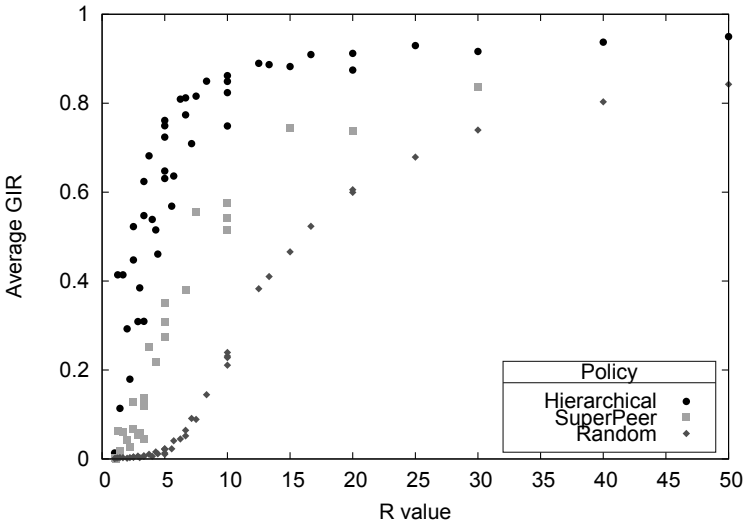



**Fig. 2.** Mean GIR values for the three studied policies: Hierarchical, Super-peer and Random. Each point represents the mean GIR value of a complete simulation.

a similar behavior in this scenario: the nodes tend to have only the information about their own resources resulting in low GIR values.

On the other hand, high values of R produce an overall good performance in all the policies, although Random presents worse performance even in the case of very high R values.

The middle values of R present the substantial part of the system behavior: Hierarchical presents the best performance, showing a sharply increment with R and remaining almost constant after a value of 10. Super-peer has a smooth increase in performance with R, keeping the improvement over all the range of studied R. Finally, Random presents the lower GIR values, slightly improving with the increase of R.

Looking at fixed values of R in the medium range (i.e. R= 15), the relation is maintained: Hierarchical first, followed first by Super-peer and then by Random.

The slope in the obtained GIR value can be used to determine the limit R value that can be set to maintain the system updated: in the case of Hierarchical policy, increasing the value of R more than 10 has no additional impact, i.e. configuring the expiration 10 times the refresh time ensures a good system performance. Super-peer and Random policies do not show such a clear limit, increasing R keeps the improvement in resulting GIR.

### 3.2 Impact of Generated Logical Structure in Hierarchical Policy

In the case of the Hierarchical policy, a logical structure (i.e. the hierarchy) must be established beforehand. In the case of a three levels hierarchy, it can be divided in:

– the first level only includes the root node, which is any of the nodes in the network infrastructure.
– the second level is composed of the nodes directly connected to the root through the same router, i.e. the path connecting these nodes and the root has only one hop.
– the rest of the nodes are included in the third (and last) level.

This organization does not take advantage of the characteristics of the underlying network infrastructure. Having three levels means that the information exchange between two nodes in the third level may possible involve traveling a much longer path than going directly between them if a direct link exists.

In this section, we study the impact of creating a three level hierarchy starting from different nodes (which will become the root of the defined hierarchy). We use the introduced `R` metric to obtain some general conclusions about the impact of selecting the root node.

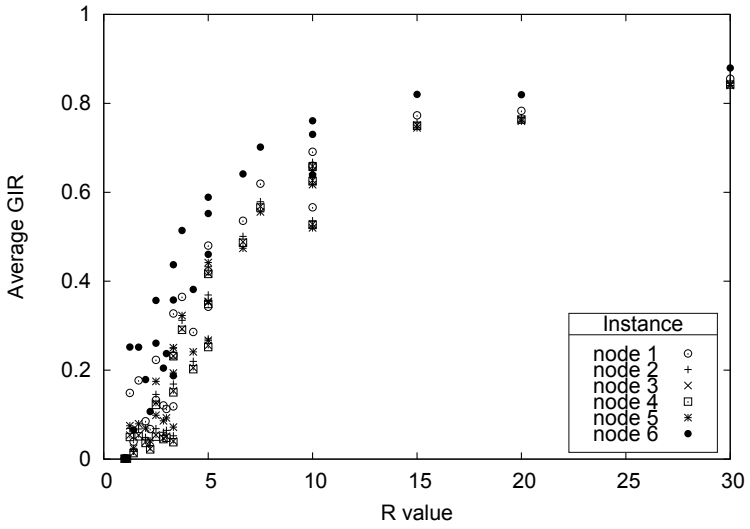



**Fig. 3.** Behavior of Hierarchical policy using different nodes to start the generation of the hierarchy. The selection of the root node is important, obtaining a consisting best performance when the node selected is the number 6. It is interesting that the relative position of the GIR values using each point is maintained in all the studied range of `R`.

Figure 3 shows the behavior of the Hierarchical policy in terms of the selected node as the root of the hierarchy. Six nodes that produce different system behavior are selected to be displayed.

The selection of the root node is important, obtaining a consisting best performance when the node selected is the number 6. It is interesting to note that the curve described by the $R$ parameter is shifted depending on the different generated hierarchies, but the relative position between them does not change.

### 3.3 How Many Super-Peers?

The logical structure of super-peer network has two levels:

- Normal nodes level: a selected node is chosen to act as the root of a one-level hierarchy, which is denominated *super-peer*. The communication in this subset of the network is maintained between the nodes and the selected super-peer, concentrating the information in this node.
- Super-peer level: this level is composed of all the super-peers in the network, exchanging the information with other super-peers. A super-peer can only communicate with other super-peers and with the normal nodes in their own partition, can not exchange information directly with normal nodes in other subset.

The amount of partitions may have impact in the system behavior: one limit case is having one node per partition (i.e. all the nodes are super-peers with no normal nodes attached); the other limit case is to have only one partition (i.e. the same distribution as a one-level hierarchy, only one super-peer is present).

The figure 4 shows the mean GIR values for the same network infrastructure varying the amount of partitions (i.e. super-peers). Random policy is included to serve as a comparison as the degenerated Super-peer policy becomes similar to it.

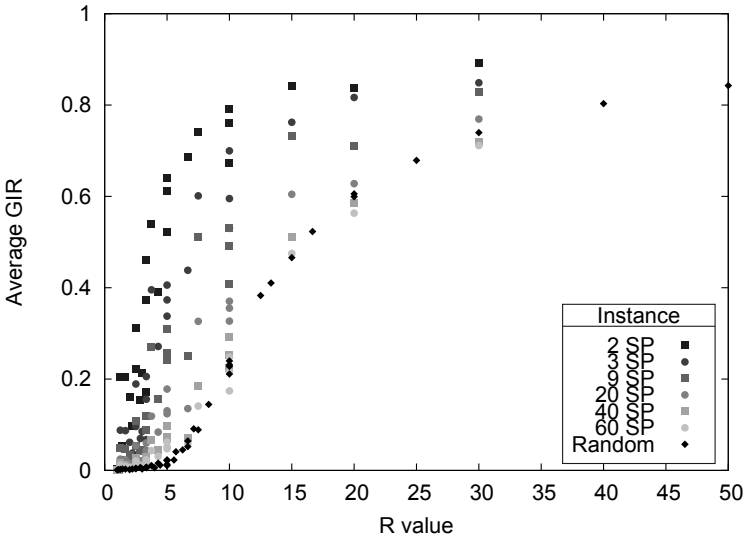



**Fig. 4.** Mean GIR values for the same network infrastructure varying the amount of super-peers. Increasing the partitions produces worse performance, Random policy and two super-peers (2-SP) act as the limiting cases in the studied `R` range.

With the increase in the number of super-peers, the system shows a worse performance. The `R` value also has impact on the peak GIR and saturation value

(i.e. the value from which the system does not show a considerable improvement with the increment of R). 2-SP saturates near R= 10, but 60-SP does not show a saturated state, which means that increasing the value of R produces a rise in the obtained GIR value.

The intermediate cases show a smooth decay in the performance with increment in the amount of super-peers in the system. The saturation point also moves to higher values of R in relation with the present super-peers, i.e. the relation between refresh and expiration times must be larger to reach the same performance. This situation translates in the need of using more resources to communicate more often between nodes to maintain the same level of global performance.

## 4 Conclusions

One of the key features needed in Grid Computing is to maintain up-to-date information about resources and services in a highly heterogeneous and dynamic environment. Using a manually established structure to obtain this is information can become a bottleneck, becoming necessary to design new policies for discovery and propagation of resource and monitoring information.

Setting up a propagation policy involves determining the way in which a node *talks* to a partner: the information refresh period and resource expiration time. We introduced R, which relates these parameters, to study three resource information policies: Hierarchical, Super-peer and Random.

The three considered policies show a similar behavior in the studied scenario: A low R value means that the nodes tend to have only the information about their own resources resulting in low GIR values. On the other hand, high values of R produce an overall good performance in all the policies, although Random presents worse performance even in the case of very high R values. The middle values of R present the substantial part of the system behavior: Hierarchical presents the best performance, showing a sharply increment with R and remaining almost constant. Super-peer has a smooth increase in performance with R, keeping the improvement over all the range of studied R.

The slope in the obtained GIR value can be used to determine the limit R value that can be set to maintain the system updated, which acts as a *saturation point*: no further improvement is obtained communicating more often.

The Hierarchical policy is based in a logical structure that must be established beforehand. The creation of such structure is started from an initial node which is selected as the *root* of the hierarchy. Based on R, the impact of creating a three level hierarchy starting from different nodes was study. The results show that the selection of the root node changes the performance of the Hierarchical policy. The curve described by the R parameter is shifted depending on the different generated hierarchies, but the relative position between them does not change.

The Super-peer policy uses a two level hybrid approach: one level is composed of super-peers using Random policy to communicate between them. Every super-peer has a subset of nodes (not super-peers) which can only communicate with

it. These nodes compose the second level. We use R value to analyze the impact of selecting different amount of super-peers in a fixed network infrastructure. The amount of partitions have impact in the system behavior: one limit case is having one node per partition (i.e. all the nodes are super-peers with no normal nodes attached); the other limit case is to have only one partition (i.e. the same distribution as a one-level hierarchy, only one super-peer is present). The intermediate cases show a smooth decay in the performance with increment of super-peers in the system. The saturation point also moves to higher values of R in relation with the present super-peers: the relation between refresh and expiration times must be larger to reach the same performance.

The R resulted very useful to analyze the global behavior of the system and can provide very useful information as the saturation point. The study of this new parameter is essential to provide a deeper knowledge in new information distribution policies and in the configuration or deployment of existing ones.

## Acknowledgments

E.M. is investigator of the CONICET. This work was partially supported by grants from Universidad de Buenos Aires (UBACyT 20020090300030) and CONICET (PIP 1087/09).

## References

[1] Globus Alliance web page, http://www.globus.org/, last visited on 02/11/2011
[2] MDS in Globus Alliance, http://www.globus.org/toolkit/mds/, last visited on 02/11/2011
[3] Abbas, A.: Grid Computing: A Practical Guide to Technology and Applications. Charles River Media, Hingham, MA (2003)
[4] Aloisio, G., Cafaro, M., Epicoco, I., Fiore, S., Lezzi, D., Mirto, M., Mocavero, S.: Resource and service discovery in the igrid information service. In: Computational Science and Its Applications - ICCSA. pp. 1–9 (2005)
[5] Casanova, H., Legrand, A., Quinson, M.: Simgrid: A generic framework for large-scale distributed experiments. In: 10th IEEE International Conference on Computer Modeling and Simulation. pp. 126–131. IEEE Computer Society, Los Alamitos, CA, USA (Mar 2008)
[6] Czajkowski, K., Fitzgerald, S., Foster, I., Kesselman, C.: Grid information services for distributed resource sharing. In: HPDC '01: Proceedings of the 10th IEEE International Symposium on High Performance Distributed Computing. pp. 181–194. IEEE Computer Society, Washington, DC, USA (2001)
[7] De Roure, D., Baker, M., Jennings, N., Shadbolt, N.: Grid computing - making the global infrastructure a reality, chap. The evolution of the Grid, pp. 65–100. John Wiley & Sons Ltd (2003), http://eprints.ecs.soton.ac.uk/6871/
[8] Foster, I., Kesselman, C., Nick, J., S., T.: The physiology of the grid: An open grid services architecture for distributed systems integration. Tech. rep., Open Grid Service Infrastructure WG, Global Grid Forum (2002), http://www.globus.org/research/papers/ogsa.pdf

[9] Foster, I.: Globus Toolkit Version 4: Software for Service-Oriented Systems. In: Jin, H., Reed, D., Jiang, W. (eds.) IFIP International Conference on Network and Parallel Computing 2005. Lecture Notes in Computer Science, vol. 3779, pp. 2–13. Springer Berlin / Heidelberg (2005)

[10] Foster, I., Kesselman, C.: The Grid 2: Blueprint for a New Computing Infrastructure. The Morgan Kaufmann Series in Computer Architecture and Design, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (Nov 2003)

[11] González Márquez, D., Mocskos, E.E., Fernández Slezak, D., Turjanski, P.G.: Simulation of resource monitoring and discovery in grids. In: Proceedings of HPC 2010 High-Performance Computing Symposium. pp. 3258–3270 (2010), http://www.39jaiio.org.ar/node/121

[12] Iamnitchi, A., Foster, I., Nurmi, D.: A peer-to-peer approach to resource discovery in grid environments. In: Proceedings of the 11 th IEEE International Symposium on High Performance Distributed Computing HPDC-11 (HPDC' 02). p. 419. IEEE, Edinbourgh, UK (Jul 2002)

[13] Iamnitchi, A., Foster, I.: Grid resource management: state of the art and future trends, chap. A peer-to-peer approach to resource location in Grid environments, pp. 413–429. Kluwer Academic Publishers, Norwell, MA, USA (2004)

[14] Massie, M.L., Chun, B.N., Culler, D.E.: The ganglia distributed monitoring system: design, implementation, and experience. Parallel Computing 30(7), 817–840 (Jul 2004), http://www.sciencedirect.com/science/article/pii/S0167819104000535

[15] Mastroianni, C., Talia, D., Verta, O.: A super-peer model for resource discovery services in large-scale Grids. Future Generation Computer Systems 21(8), 1235–1248 (October 2005)

[16] Mastroianni, C., Talia, D., Verta, O.: Designing an information system for grids: Comparing hierarchical, decentralized P2P and super-peer models. Parallel Computing 34(10), 593–611 (2008)

[17] Plale, B., Jacobs, C., Jensen, S., Liu, Y., Moad, C., Parab, R., Vaidya, P.: Understanding grid resource information management through a synthetic database benchmark/workload. In: CCGRID '04: Proceedings of the 2004 IEEE International Symposium on Cluster Computing and the Grid. pp. 277–284. IEEE Computer Society, Washington, DC, USA (Apr 2004)

[18] Puppin, D., Moncelli, S., Baraglia, R., Tonellotto, N., Silvestri, F.: A grid information service based on peer-to-peer. In: Cunha, J.C., Medeiros, P.D. (eds.) Euro-Par 2005 Parallel Processing. Lecture Notes in Computer Science, vol. 3648, pp. 454–464. Springer (2005)

[19] Ranjan, R., Harwood, A., Buyya, R.: Peer-to-peer-based resource discovery in global grids: a tutorial. Communications Surveys & Tutorials, IEEE 10(2), 6–33 (2008)

[20] Trunfio, P., Talia, D., Papadakis, C., Fragopoulou, P., Mordacchini, M., Pennanen, M., Popov, K., Vlassov, V., Haridi, S.: Peer-to-Peer resource discovery in Grids: Models and systems. Future Generation Computer Systems 23(7), 864–878 (Aug 2007)

[21] Yang, B., Garcia-Molina, H.: Designing a Super-Peer Network. In: Proceedings of 19th International Conference on Data Engineering (ICDE'03). p. 49. IEEE Computer Society, Los Alamitos, CA, USA (2003)