Benchmarking Grid Information Systems

Laurence Field[1] and Rizos Sakellariou[2]

[1] CERN, Geneva, Switzerland
Laurence.Field@cern.ch
[2] The University of Manchester, Manchester, UK
rizos@cs.man.ac.uk

Abstract. Grid information systems play a central role in today's production Grid infrastructures, enabling the discovery of a range of information about the Grid services that exist in an infrastructure. As the number of services within these infrastructures continues to grow, it must be understood whether the current implementations are able to scale to meet the future requirements. Existing approaches for evaluating Grid information systems mainly focus on performance metrics and do not consider the quality of the information itself. This paper proposes a comprehensive benchmarking methodology for the evaluation of Grid information systems which includes a metric to assess the quality of the information returned. Using this methodology, two commonly used Grid information system implementations, Metadata Directory Service (MDS) and the Berkeley Database Information Index (BDII), are evaluated using data obtained from the Enabling Grids for E-SciencE (EGEE) production Grid.

1 Introduction

Grid information systems enable users, applications and services to discover which Grid services exist in a Grid infrastructure along with further information about their structure and state [2,7]. Information describing each Grid service originates from the Grid service itself and hence the Grid service, in terms of Grid computing, is the primary information source. Grid information systems provide an interface that can be used to resolve queries over these information sources, which can be numerous and widely distributed geographically. From the perspective of an information consumer, the information system can be represented by an abstract interface to which queries can be sent and a query response is received. How the Grid information system resolves that query, taking into consideration all the information sources, is an implementation detail for the specific system. What matters from the users perspective is the overall quality of the service provided by the interface.

As Grid infrastructures grow, more Grid services and hence information sources will exist in the infrastructure, which will increase the total volume of information. In addition, a larger infrastructure also experiences greater utilization and the amount of queries made to the information system will also increase. Ensuring that Grid information systems will meet the future requirements in

E. Jeannot, R. Namyst, and J. Roman (Eds.): Euro-Par 2011, LNCS 6852, Part I, pp. 479–490, 2011.

terms of information volume and query load, is one of the main challenges facing today's production Grid infrastructures.

In recent years, a number of Grid information systems have been developed. Despite their differences [4], there is a great deal of commonality between their core functionality and deployment scenarios. Grid infrastructure managers need to compare these different Grid information systems implementations in order to choose which solution best meets the needs of their infrastructure. This is a decision that should not be taken lightly. Large scale distributed infrastructures can exist for many years, during which time it can be too difficult to migrate between different implementations. A benchmarking methodology for evaluating and comparing these different implementations would provide valuable insight for Grid infrastructure managers. Such a methodology should evaluate the Grid information system in order to understand its scalability limitations. The results of this evaluation could then be compared with the estimated future requirements from growth projections of that Grid infrastructure.

Existing literature evaluating Grid information systems [3,9,11,12,13,14,15] focuses mainly on performance metrics such as response time (that is, how quickly a user query is answered) or throughput (that is, how many concurrent queries can be handled over a period of time). However, these metrics alone cannot be used to assess the overall performance of Grid information systems as they neglect some important aspects such as the nature of the queries (different queries have different requirements) as well as the quality of the answers delivered by the Grid information system. In fact, a recent study [5], whose results largely motivated the present paper, demonstrated that how up-to-date the information delivered by a Grid information system is may vary significantly across different types of Grid information; this affects the quality of the answers obtained and must be considered in any evaluation study.

Taking into account the above, this paper presents a comprehensive benchmarking methodology to evaluate Grid information systems, which is based on the assessment of three different components: (i) a set of commonly executed queries; (ii) query response time; and (iii) quality of information returned. To the best of our knowledge, this is the first time that such a comprehensive benchmarking methodology for Grid information systems, which incorporates the use of a *quality metric*, is proposed. The proposed benchmarking methodology is then used to evaluate two different Grid information system implementations; the Metacomputing Directory Service (MDS) [2] from the Globus project and the Berkeley Database Information Index (BDII) [6], which is a simplified implementation of Metadata Directory Service (MDS). The evaluation takes place on the Enabling Grids for E-science (EGEE) [8] infrastructure, the largest multidisciplinary Grid infrastructure in the world.

The paper is outlined as follows. Section 2 gives a critical evaluation of previous, related work. Section 3 describes in detail the components of the benchmarking methodology and how the metrics included can be measured. The evaluation

of the two Grid information systems using the benchmarking methodology presented is given in Section 4. Finally, some concluding remarks can be found in Section 5.

2 Related Work

This section reviews work that has been done to evaluate Grid information systems. As already mentioned, and will be observed next, most of this work does not assess the quality of the information obtained.

In a frequently cited study [13,14,15], the functional components of three systems were grouped into similar types and the performance of each component type was evaluated against those of their counterparts. The focus was to evaluate the effect of a large number of users and information sources for each implementation. In addition to the average query response time, a throughput metric (average number of queries per second) was also used. Both of these metrics were measured for different numbers of concurrent queries and information sources. However, in this study little attention was given to the data itself. A single query (return all information for a specific information source) was used and it was assumed that the response was always *correct* (i.e., up-to-date, or current, or fresh). In addition, the information volume used was small (10Kb) compared with today's infrastructures.

A study investigating the scalability and analyzing the performance of an information system [3] addressed some of these deficiencies by using real data from a large-scale Grid infrastructure. It also investigated how the performance is affected by the size of the query response, however, again only the average query response time was used as a metric and again it was assumed that all responses were correct.

A Grid information benchmark has previously been proposed [9], based on a set of queries and scenarios, which were used to compare the access language and platform capabilities for three different database platforms serving Grid information. Although in total 13 different queries were used, the specific case for using this set was not made. The databases were populated with synthetic but realistic information about Grid services, which conformed to the GLUE information model [1]. Short synthetic workloads (scenarios) were used to measure the query response time of concurrent query requests to the repository. The major difference with our work is that we include a quality metric and actual results from a production Grid.

Of particular interest is the study in [12], which, in addition to query response time, proposed two further metrics, network overhead and information freshness, for evaluating Grid information systems. Network overhead is defined as the number of bytes that a crawler downloads to update the information in the system. The crawler in this context is a specific implementation detail of this system and although important, it is not a concern for the user from a quality of service perspective. Information freshness, on the other hand is of great concern to the user. Information of the Grid information system is considered fresh (or

correct), at a given point in time, if it is synchronized with its real-world equivalent value at the information source (that is, both values are the same). This concept of freshness is useful as a quality metric for the information returned, and it is missing from other studies. However, the proposed calculation of freshness may not be feasible in a real system as it requires a continuous comparison of all values.

All of the above studies demonstrate the need to evaluate Grid information systems. They show that the query response time is a key metric and should be a core part of any proposed benchmarking method. The studies also show that the query response time can be affected by four main factors; the total information volume, the type of query (both due to the complexity and size of the response), the number of concurrent queries (query loading) and the implementation (hardware, software and internal architecture).

Extending the current state-of-art further, our benchmarking methodology takes into account the above, also including a metric to measure the quality of the query response.

3 Methodology

The benchmarking methodology will be based on the GLUE 1.3 information model [1] as this information model is used by the majority of today's production Grid infrastructures [10]. The information model used is a key part of any benchmarking study as it defines a number of constants of relevance to the benchmarking method. These include the object types, and hence expected frequency of change for those object types, the composition of the queries and the size of the query response. If an alternative information model is used, these constants will have to be measured for that information model. Due to the different constants being used, benchmarking results cannot be compared for different information models.

The volume of information and the use case for the benchmarking methodology will be taken from a large-scale production Grid infrastructure, the infrastructure from the Enabling Grids for E-science (EGEE) project. This is a fair choice to use as a reference for the benchmarking methodology as EGEE operates the largest multi-disciplinary Grid infrastructure in the world. In addition, it also makes use of the GLUE 1.3 information model. As of November 2010, the EGEE Grid information system contained information representing 4102 Grid services deployed between 375 sites. This information represents an information size of 102MB in the LDIF format, which corresponds to an average of 272KB per site or 25KB per service.

Set of queries: In order to consider queries commonly made to the EGEE Grid information system, we used the information provided in [3] to choose a top-ten of queries. The type of each query and its frequency as a percentage of the total number of queries made to the system are shown in Table 1.

We further analyzed the characteristics of each query in terms of the query response size and the object type that they use. As different objects experience

Table 1. The top ten queries made to the EGEE infrastructure as a percentage of the total number of queries

	Query	%
Q1	Find the Closest Computing Service to a Storage Service	19.6
Q2	Find the VO's Storage Area for a Storage Service	17.7
Q3	Find all Storage Services	16.3
Q4	Find a Storage Service	15.5
Q5	Find the Closest Storage Service to a Computing Service	7.8
Q6	Find all Services for a VO	6.8
Q7	Find all Computing Services for a VO	2.1
Q8	Find all Storage Areas for a VO	2.1
Q9	Find all Sub Clusters	1.5
Q10	Find the VO's Computing Share for a Computing Service	1.4

a different frequency of change [5], the object type being used by each query may indicate how up-to-date the results of this query are expected to be. These characteristics are shown in Table 2.

A few observations are interesting to note. All queries have a similar level of complexity: they either return all the objects of a particular type or filter by only one predicate. As such, the two key differences between the ten queries are the size of the query response and the object type that is queried.

The size of the query response allows a classification of the queries into three groups: (i) queries that return information about one service (denoted by 'small' in the table); (ii) queries that return all information for an object type (denoted by 'large' in the table); and (iii) queries that filter information for one object type (denoted by 'varying' in the table). For queries of the latter group the actual size of the response varies depending on the filter being used. To ensure that the filter used for a particular query provides a good representation of the query response size for that query, all possible values for that filter were evaluated and

Table 2. Characteristics of each of the top ten queries

Query	Size(bytes)	Size(class.)	Object	Frequency
Q1	2858	small	CESEBind	Low
Q2	5436	small	SA	High
Q3	315225	large	SE	High
Q4	642	small	SE	High
Q5	1875	small	CESEBind	Low
Q6	49801	varying	Service	High
Q7	16607	varying	CE	High
Q8	12348	varying	SA	High
Q9	8959015	large	SubCluster	High
Q10	6009	varying	VOView	High

the response size in each case was calculated. The median size was then chosen (and is shown in the table) to avoid the average being skewed by the few large query responses. Regarding the object type used by each query, we classify these objects according to the classification in [5] into high-frequency objects (those that experience over 1% changes per day) and low-frequency objects (those that experience less than 1% changes per day).

Query response time: Each query is executed 50 times against the deployed instance of the Grid information system under evaluation. Each time, the query response time (QRT) is measured from the start of the client connection to when all the data for the query response has been received by the client. The average query response time can be calculated using Equation 1, where $n = 50$.

$$QRT_{average} = \frac{\sum QRT}{n} \tag{1}$$

Quality of information: For each query, the (α, β)-currency will be used to assess the quality of information returned. This metric, first investigated in the context of the Grid in [5], can be used to calculate a probability, α, that a selected object value stored by the Grid information system is current with respect to a grace period β. The probability α can be calculated by using Equation 2, where β is the age of the information and λ is a constant for each object type.

$$\alpha = e^{-\lambda\beta} \tag{2}$$

The method used to measure β, the age of the information, depends on the specific Grid information system implementation. If a timestamp is available for when the information was created, this can be compared to the time when the query was executed. Alternatively, the time between when the information was created and when the query was executed will need to be measured. The latter will be the method that will be used later in our benchmarking comparison of MDS and BDII. We assume that the information is fresh and the query is executed as soon as the cache has been updated. Hence, we use the time it takes to update the cache as the time between when the information was created and when the query was executed; this provides a value for β.

The constant λ can be calculated using Equation 3, where f is the frequency of change for a specific object type and N is the number of objects of that type,

$$\lambda = \frac{f}{N}. \tag{3}$$

Further information regarding the frequencies of change for the GLUE 1.3 object types (as well as the number of objects for each type) can be found in [5].

4 Benchmarking MDS and BDII

This section benchmarks two Grid information system implementations, MDS and BDII, using the benchmarking methodology described in the previous section.

4.1 Background

The MDS query interface hides the details [2,7] of the MDS architecture and implementation from the client. This interface is provided in MDS by the Grid Index Information Service (GIIS) using the *Grid Resource Information Protocol* (GRIP). The GIIS maintains a dynamic registry for information source locations populated from registration notifications received via the *Grid Resource Registration Protocol* (GRRP). The GIIS parses each incoming GRIP request and then forwards this request to one or more information sources found in the registry depending on the type of information requested. To improve performance [15], each response may be cached for a configurable period of time by specifying the cache time-to-live (TTL) per information source as part of the registration. In the MDS implementation, the standard Lightweight Directory Access Protocol (LDAP) has been adopted for both the GRRP and GRIP, where it was used to define the data model, query language and wire protocol, and the GIIS is implemented as a special purpose back-end for an OpenLDAP server.

The Berkeley Database Information Index (BDII) is a simplified implementation of the MDS GIIS. The main simplification is the absence of the GRRP protocol with the dynamic registry functionality being replaced by a static registry where new information source locations are added manually by a system administrator. The other simplification is that the BDII maintains its information cache using a parallel process rather than caching query results. This serves to decouple the cache update process from incoming queries; therefore, the information sources are isolated from the queries and this results in a predictable behavior of the system.

In both implementations, the information is cached and queries are evaluated using this cache. The implementations diverge in the method employed to refresh that cache. Further discussion on the impact of the cache update method can be found in Section 4.5.

4.2 Experiment Setup

The MDS and BDII software was installed and configured on a physical machine. The machine used had a 1.80GHz single processor Intel Pentium and 1Gb of memory. The MDS version used was 2.4, which is based on OpenLDAP 2.0 and the BDII version used was 5.1.9, which is based on OpenLDAP 2.3. A second machine was used to run client queries. This machine had four 2.66GHz Intel Xeon CPUs and 8Gb of memory. The two machines were connected through a 100 Mbps LAN.

The caches were populated with data representing the information from the EGEE production infrastructure and the cache update processes were disabled.

4.3 Query Response Time

Each query was executed in a single thread that iterated 50 times over the same query. The average response time for each query type made to the MDS GIIS and the BDII can be seen in Figure 1.

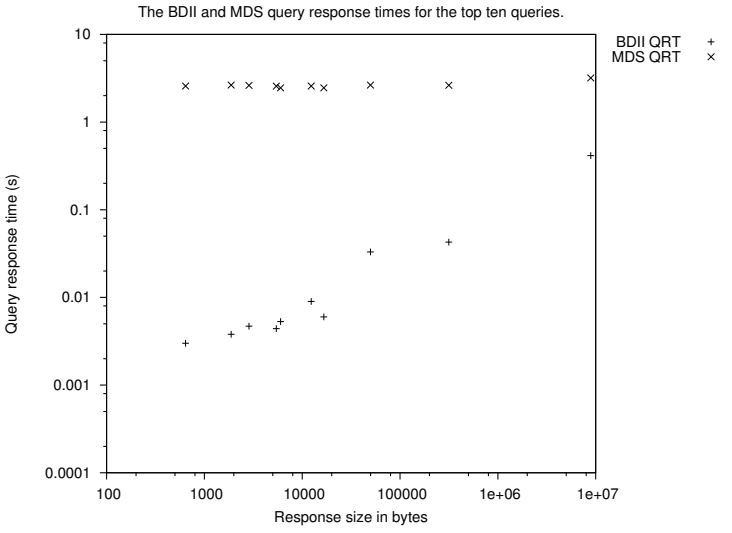

Fig. 1. Average query response time for the top ten queries made to MDS and BDII

These results show that for MDS GIIS the query response time is almost constant, approximately 2.6s, for all query result sizes, with the exception of the 8.9Mb result size, which causes the query response time to increase to 3.2s. This suggests that the connection overhead and query execution time are the main factors that affect the query response time for the MDS GIIS. The query result size, and hence transfer overhead, only has an impact for large query response sizes. By contrast, the BDII implementation has much lower query response times, which seem to be affected by the query response size. This suggests that the connection overhead and query execution time are much lower for the BDII implementation and that the transfer overhead is the main factor that affects the query response time.

4.4 Quality of Information

As already mentioned in Section 3, for the value of β we can use the time it takes to update the cache. However, we note that the BDII and MDS GIIS differ in the way the cache is updated. In the MDS GIIS, the cache is updated synchronously with the query, whereas with the BDII this update is done asynchronously. This means that for the MDS GIIS, when the cache is stale, the next query will trigger the update process and subsequent queries will block until this process has finished. In the BDII a parallel process updates the cache while the cache is still responding to queries. In either case, the time to update the cache is the minimal value for β in Equation 2, with the maximum value being defined by the TTL of the cache in the MDS GIIS or the delay between update cycles in the BDII. The best-case scenario, which will be measured here, is where these values are zero.

Table 3. The result of calculating (α, β)-currency for the MDS GIIS and BDII

Query	Object	λ	α_{MDS}	α_{BDII}
Q1	CESEBind	1.14×10^{-08}	1.000	1.000
Q2	SA	5.11×10^{-04}	0.279	0.902
Q3	SE	4.85×10^{-04}	0.298	0.907
Q4	SE	4.85×10^{-04}	0.298	0.907
Q5	CESEBind	1.14×10^{-08}	1.000	1.000
Q6	Service	4.86×10^{-04}	0.298	0.907
Q7	CE	1.91×10^{-03}	0.008	0.681
Q8	SA	5.11×10^{-04}	0.279	0.902
Q9	SubCluster	2.28×10^{-06}	0.994	1.000
Q10	VOView	1.48×10^{-03}	0.025	0.743

To measure the query response time for the MDS GIIS the cache TTL was set to zero so that every query made to the cache would trigger the update process. A simple query that returned zero results was used to trigger the update. The query response time for 10 successive queries was measured and the average query response time was calculated. The average query response time when the cache is valid was also calculated. The difference between the query response time when the cache is valid and invalid gives the cache update time. The cache update time for the MDS GIIS was 2496s with σ=13.6s.

For the BDII, the update process is instrumented and the update time is recorded in the log file of the update process. Thus, the average time for 10 updates was calculated. The cache update time for the BDII was 201s with σ=14.9s.

Using the cache update times for β (that is, β_{MDS}=2496 and β_{BDII}=201) and the frequencies of change from [5] to calculate λ, the value of α was calculated using Equation 2 for each query. The results are shown in Table 3. The values α_{MDS} and α_{BDII} correspond to the probability that the information is current for the MDS GIIS and BDII, respectively.

4.5 Discussion

Comparing the query response times of the BDII and MDS GIIS implementations for a single query thread, the result depends on the query used. For Q4, the query with the smallest response size (642 bytes), the query response times are 0.003s and 2.58s for the BDII and the MDS GIIS, respectively. For Q9, the query with the largest response size (8.5Mbytes), the query response times are 0.41s and 3.19s, respectively. The BDII implementation delivers much better performance than the MDS GIIS for queries with a small response size, however, this advantage is reduced for larger response sizes.

As stated previously, looking at the query response time is only one aspect of the overall quality of service. The quality of the information returned also needs to be considered. In the case of querying a low frequency object type, for example Q1 (CESEBind), it is almost certain that all information is current.

However, for the case of querying a high frequency object type, Q7 (CE), the probability that the information is current is 0.008 for the MDS GIIS, while, for the BDII, the probability that the information is current is 0.681. We note that these figures represent the best-case scenario as they assume the original information is current. The value for α will degrade further during the period after the caches have been updated. The length of this period can be defined by a configuration parameter in both instances. Knowledge of (α, β)-currency can be used to set an optimal value for this parameter, however, the limit is reached when the value is set to zero.

Comparing the query response times will show which implementation gives a faster response, however, it does not indicate whether the query response time is acceptable. In both implementations, the query response time may be acceptable for a particular deployment scenario. Including the (α, β)-currency metric gives additional insight by providing a measurement of the quality of the information returned. What values are acceptable will be dependent on the particular deployment scenario, however, we can make some general statements about the result. Any value less than 1 would result in the client having to include a probabilistic approach when using this information. Any value less than 0.5 means that the information returned is more likely to be incorrect than it is likely to be correct. With this interpretation, the MDS GIIS is returning more incorrect information when querying high frequency object types than correct information and although the BDII is returning more correct information than incorrect information when querying high frequency object types, a probabilistic approach to using that information would be required.

We also note that when the cache in the MDS GIIS is stale, the next query will trigger the cache to be updated and subsequent queries will be blocked until this process has finished. Unlike the MDS GIIS, the BDII can be queried while the cache is being refreshed. To see how the query load affects the cache update time and hence β, the query Q3 was used to produce a query load using a single thread. This query load caused the cache update time to increase to 1100s with σ=145 while the average query response time for the query increased from 0.0427s with σ=0.00114 to 0.0472s with σ=0.0339. Although for low frequency object types, it is still almost certain that the information in the cache is current (which implies better quality), the value of α for high frequency object types was reduced from 0.907 to 0.58 (for Q3). Also, the values for Q7 and Q10 were reduced from 0.681 and 0.743 to 0.12 and 0.2, respectively.

As a final remark, we note that a potentially important aspect for Grid information system benchmarking, which has not been a central issue in this paper, but in other similar studies, is how the query response time varies with the number of parallel queries. In some preliminary results, the query response time for varying numbers of concurrent queries executed against the MDS GIIS and BDII was measured. The results showed that for the MDS GIIS doubling the number of query threads essentially doubles the query response time. However, with only 4 query threads, a few queries time out, that is they exceed the 20s threshold set

in the test framework as a protection against queries hanging indefinitely. With only 6 query threads, up to 35% of the queries time out and with 8 query threads up to 52% of the queries time out. For the BDII up to 100 parallel query threads were used, as provisional tests showed that BDII could handle much more than 8 query threads. Even with 100 query threads, many of the queries return a result in less than 1s. The exception is for queries that return large query responses, where timeouts were observed with 75 and 100 query threads. Although briefly mentioned here, the issue of concurrent query execution is highly relevant for today's production Grid infrastructures; however, lack of space does not allow us to cover it in-depth and has been covered in other related studies.

5 Conclusion

This paper investigated the benchmarking of Grid information systems and proposed a comprehensive benchmarking methodology, suggesting that the addition of (α, β)-currency as a metric for describing the quality of a Grid Information System would give additional insight. Two Grid information system implementations that have had production exposure, MDS and the BDII, were then benchmarked using this methodology. The results showed that in terms of response time, the BDII implementation gives better performance than the MDS GIIS for queries with a small response size, 0.003s and 2.58s for the BDII and the MDS GIIS respectively for Q4 (642 bytes). However, this advantage is reduced for larger response sizes, giving 0.41s and 3.19s for the BDII and the MDS GIIS, respectively, for Q9 (8.5Mbytes).

While this comparison showed that the query response time for the BDII was better (faster) than for the MDS GIIS, a statement could not be made on whether or not this result was acceptable for deployment in a production Grid infrastructure. Including the (α, β)-currency metric gave additional insight by providing a quality measurement for the information returned. For low frequency object types, it is almost certain that the information in the cache is current. However, for the high frequency object type in Q7 (CE), the probability that the information is current is 0.008 and 0.681 for the MDS GIIS and the BDII, respectively. This suggests that for Q7, the MDS GIIS would mainly be returning incorrect values and that a probabilistic approach would be required for the information returned from the BDII.

In conclusion, this paper has presented a comprehensive methodology for benchmarking Grid information systems. This methodology demonstrated that including the concept of (α, β)-currency gives additional insight over using the query response time alone. Our experimental results have also indicated that the existing Grid information system used in the EGEE production infrastructure is not particularly suited for high frequency object types. Future work can consider further improvements of the query set or more exhaustive experimental results.

References

1. Andreozzi, S., Burke, S., Field, L., Litmaath, M.: GLUE schema version 1.3, http://glueschema.forge.cnaf.infn.it/Spec/V13
2. Czajkowski, K., Fitzgerald, S., Foster, I., Kesselman, C.: Grid information services for distributed resource sharing. In: Proceedings of the 10th IEEE International Symposium on High Performance Distributed Computing, San Francisco, CA, USA, pp. 181–194 (2001)
3. Ehm, F., Field, L., Schulz, M.W.: Scalability and performance analysis of the EGEE information system. Journal of Physics: Conference Series 119(6), 062029 (2008)
4. Field, L., Andreozzi, S., Konya, B.: Grid information system interoperability: The need for a common information model. In: Proceedings of the 4th IEEE International Conference on eScience, Indianapolis, IN, USA, pp. 501–507 (2008)
5. Field, L., Sakellariou, R.: How dynamic is the grid? Towards a quality metric for grid information systems. In: Proceedings of the 11th ACM/IEEE International Conference on Grid Computing, Brussels, Belgium, pp. 113–120 (2010)
6. Field, L., Schulz, M.W.: Grid deployment experiences: The path to a production quality LDAP based grid information system. In: Proceedings of the Conference for Computing in High-Energy and Nuclear Physics, pp. 723–726 (2004)
7. Fitzgerald, S., Foster, I., Kesselman, C., von Laszewski, G., Smith, W., Tuecke, S.: A directory service for configuring high-performance distributed computations. In: Proceedings of the 6th IEEE International Symposium on High Performance Distributed Computing, Portland, OR, USA, pp. 365–375 (1997)
8. Gagliardi, F., Jones, B., Grey, F., Heikkurinen, M.: Building an infrastructure for scientific grid computing: status and goals of the EGEE project. Philosophical Transactions. Series A, Mathematical, Physical, and Engineering Sciences 363(1833), 1729–1742 (2005)
9. Plale, B., Jacobs, C., Jenson, S., Liu, Y., Moad, C., Parab, R., Vaidya, P.: Understanding grid resource information management through a synthetic database benchmark/workload. In: Proceedings of the 4th IEEE International Symposium on Cluster Computing and the Grid (CCGrid), Chicago, IL, USA, pp. 277–284 (2004)
10. Riedel, M.: Interoperation of world-wide production e-Science infrastructures. Concurrency and Computation: Practice and Experience 21(8), 961–990 (2009)
11. Smith, W., Waheed, A., Meyers, D., Yan, J.: An evaluation of alternative designs for a grid information service. In: Proceedings of the 9th IEEE International Symposium on High Performance Distributed Computing, Pittsburgh, PA, USA, pp. 185–192 (2000)
12. Zanikolas, S., Sakellariou, R.: An importance-aware architecture for large-scale grid information services. Parallel Processing Letters 18(3), 347–370 (2008)
13. Zhang, X., Freschl, J., Schopf, J.: A performance study of monitoring and information services for distributed systems. In: Proceedings of the 12th IEEE International Symposium on High Performance Distributed Computing, Seattle, WA, USA, pp. 270–281 (2003)
14. Zhang, X., Freschl, J.L., Schopf, J.M.: Scalability analysis of three monitoring and information systems: MDS2, R-GMA, and Hawkeye. Journal of Parallel and Distributed Computing (JPDC) 67(8), 883–902 (2007)
15. Zhang, X., Schopf, J.: Performance analysis of the globus toolkit monitoring and discovery service, MDS2. In: IEEE International Conference on Performance, Computing, and Communications, Phoenix, AZ, USA, pp. 843–849 (2004)