A TAXONOMY OF GRID RESOURCE BROKERS[*]

Attila Kertész
Institute of Informatics, University of Szeged,
H-6721 Szeged, P. O. Box 652, Hungary
MTA SZTAKI Computer and Automation Research Institute,
H-1518 Budapest, P. O. Box 63, Hungary
CoreGRID Institute on Resource Management and Scheduling
keratt@inf.u-szeged.hu

Péter Kacsuk
MTA SZTAKI Computer and Automation Research Institute,
H-1518 Budapest, P. O. Box 63, Hungary
CoreGRID Institute on Resource Management and Scheduling
kacsuk@sztaki.hu

Abstract: Grid computing has gone through some generations and as a result only a few widely used middleware architectures remain. Using the tools of these middlewares, various resource brokers have been developed to automate job submission over different grids. Most of the present brokers operate only on a single grid infrastructure, where they have been developed. This taxonomy helps identifying and categorizing the most important properties of brokers within different Resource Management Systems. The result of this work reveals the differences of the examined Resource Brokers, which can enhance a more efficient grid usage and future development.

Keywords: Grid Computing, Taxonomy, Resource Broker, Scheduler, Grid Middleware

[*] This research work is carried out under the FP6 Network of Excellence CoreGRID funded by the European Commission (Contract IST-2002-004265)

1. Introduction

The Grid was originally proposed as a global computational infrastructure to solve grand-challenge, computational intensive problems that cannot be handled within reasonable time [1]. The first decade of grid research aimed at creating relatively reliable infrastructures to serve researchers and attract users. These attempts have led to the present grid middlewares, and now development is focusing on user requirements.

Executing a job in a grid environment requires special skills such as how to find out the actual state of the grid, how to reach the resources, etc. As the number of the users is growing and grid services have started to become commercial, resource brokers are needed to free the users from the cumbersome work of job handling. Though most of the existing grid middlewares give the opportunity to choose the environment for the user's task to run, originally they are lacking such a tool that automates the discovery and selection. Brokers meant to solve this problem [2]. As resource management is a key component of current grid middlewares, many solutions have been developed up till now. To enhance the manageability of grid resources and users, Virtual Organizations were founded. This kind of grouping started an isolation process in grid development, too. Interoperability among these "islands" will play an important role in grid research. This paper gives a classification of the present Grid Resource Brokers by the most relevant properties and functionalities. Identifying the key features and mapping them to user needs will open a new way for enhancing interoperability among different grids. Although the same services are available in different middlewares, they have been implemented in different ways. This taxonomy reflects the various ways how these brokers are built up and can be accessed. We believe that this paper helps researchers to have a better understanding of the current trends of resource brokerage.

2. Related work

Regarding taxonomies in Grid Computing, two main papers have been published about resource management systems [3] and workflow management systems [4]. As resource brokers are usually parts of some resource management systems, the first one is closer to our work. That taxonomy introduces an abstract model of resource management in different Grid Systems, then describes and compares the existing architectures. In this paper we are focusing more on smaller entities responsible for brokerage; these can be considered as higher-level tools of resource management systems. While each RMS operates on one middleware, Grid Resource Brokers move towards nearly independent entities and several able to access resources of different middlewares. This taxonomy is needed to clarify the role and usage of these brokers, and to gather and present also those ones that were out of the scope of the RMS taxonomy. We examine the interface and the implementation of these brokers to reveal their main properties.

3. Taxonomy of Resource Brokers

The aim of this taxonomy is to gather the recent Grid Brokers used in Grid Communities, highlighting their main properties and examining the differences and similarities regarding their architecture and operation. We classify the revealed properties to 7 major categories and split into 3 groups. The following subsections comment the categories of these groups.

The first group is middleware oriented (Middleware Support), the second explains the mainly job related categories (Interface, Job Model, QoS and Data Movement), finally the third deals with scheduling features (Information System Support and Scheduling Model).

3.1 Grid Middlewares

The first main category – on Figure 1 – shows the underlying infrastructures of the overviewed brokers.

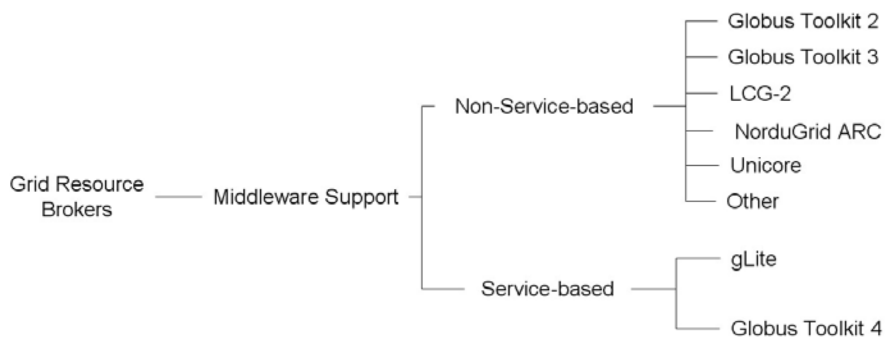

Figure 1. Categories of the Taxonomy: Grid Middlewares group

They usually rely on one of these middlewares [5][6][7][8] and use their functions to discover resources and submit user jobs. We can distinguish between service-based and non-service-based ones. Generally this property determines the architecture of the broker. It can be stated that the most widespread middleware is the Globus Toolkit, since LCG-2 is built upon Globus services and the NorduGrid ARC also uses and extends some of them.

3.2 Job handling

This group contains the mainly user and job related properties and can be seen in Figure 2.

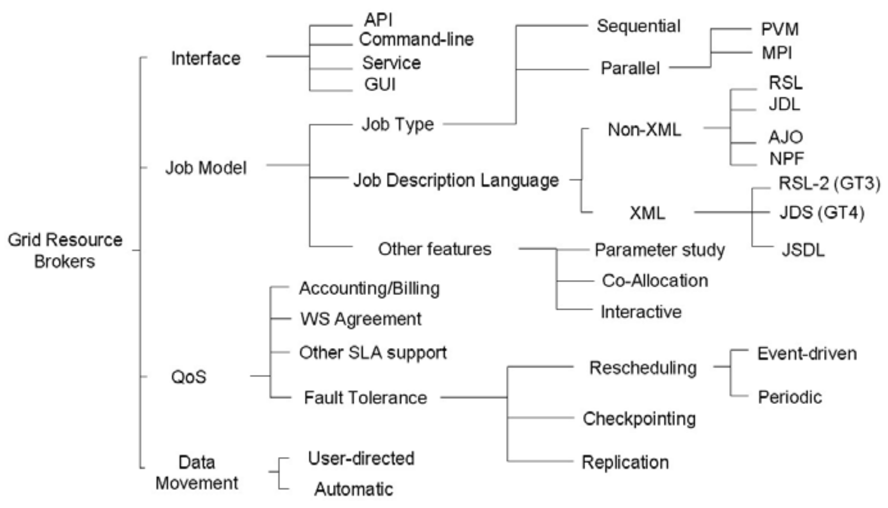

Figure 2. Categories of the Taxonomy: Job handling group

The first thing the user faces is the interface of the broker. Early solutions provided only command-line access, while APIs are important for higher level utilization and management by other applications. Some brokers even have Graphical User Interfaces to facilitate user usage. Service-based brokers offer service access, which is an advanced method and needed by the latest developments. This function can enhance interoperability and provide platform-independent access.

The job model of the broker is also important for users and applications. These properties tell how to describe a user job and what types can the broker handle. There are several non-XML language descriptions, but the latest developments follow the XML syntax. It would be reasonable for the brokers to accept and use XML job descriptions, even if they access middlewares supporting different languages [5][6][7][8]. In this case they would need to translate the request, but this approach leads to better interoperability. The rest of the properties in this subcategory show what type of jobs can be submitted with a specific broker: only sequential or parallel; in the second case co-allocation and advance reservation are handled or not. Brokers can support other special job-handling functionalities such as parameter study and interactive jobs.

Fulfilling user requirements is a critical task of the broker. Quality of Service is the collective expression of the properties to accomplish this task. Accounting is used for the administration of the users and tracking their grid utilization, and billing serves grid economy. Agreements are used to guarantee some level of service during brokering. User requirements can contain special requests, which are crucial for the job or application. On the other hand resource providers would protect sites from flooding by user jobs. In order to find a balance and fulfill requirements these policies appear in the agreements, which are taken into account in scheduling decisions. Various solutions can be developed to make these service level agreements, but this functionality is still an open issue. Basically two types are used: the WS Agreement [9] and the USLA [10]. The third part of QoS is fault tolerance. The dynamic nature of grids lowers the number of successful job submissions. To ensure a higher level of quality, brokers should be fault tolerant. Rescheduling and replication are the basic functionalities, and checkpointing can provide a more reliable brokering, though this is rarely supported, yet. Rescheduling can be event-driven or periodic, and usually choosing a different resource makes sense, retry only time consuming.

Most of the brokers provide automatic centralized data movement for input and output file staging. User-directed utilization can also be supported, when the user copies files to storage elements and tells the broker to use them.

3.3 Scheduling

The third group gathers properties related to resource information, discovery and scheduling. The properties of this group are shown on Figure 3. Several resource brokers use the information system of the underlying middleware. In this case the relevant information from the brokers' view is the data store and query. The two main subcategories are the directory-based and service-based implementations. These properties tell us how the brokers access resource data and what kind of information they can use for resource mapping – since this is determined by the information system of the middleware. Some brokers use additional information about the grid gathered by an own information system. Examining historical data (resource availability, job failures, etc.) is one of these approaches. The other type of gathering relies on agents, which provide information about specific elements of the grid.

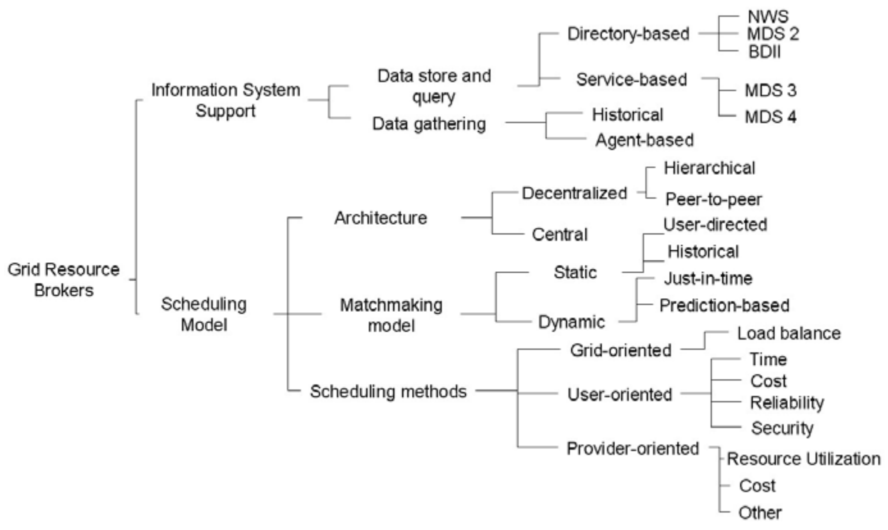

Figure 3. Categories of the Taxonomy: Scheduling group

Matchmaking is the major task of Grid Brokers. The scheduling properties can qualify brokers and determine the goodness of their decisions. In smaller scope of resources like VOs, usually a centralized scheduler component is used to make decisions. In decentralized schedulers the matchmaking process can be split up and queues can be utilized for job requests, or more components can collaborate to utilize a wider range of resources. The first solution is rather used in hierarchical and the second in peer-to-peer architectural models. The decision making can be static and dynamic. When a user fixes a resource for its job, or the scheduler component of the broker uses only static historical information, we are talking about static matchmaking. In a dynamic decision the broker has an up-to-date information about the resources and makes a just-in-time matching, or uses up some additional prediction-based information. The schedulers can take into account specific policies that affect decision making. These methods usually favor the users, but the provider expectations or the balanced state of the grid can also be observed. User policies can tell the broker to submit the job to a resource that completes the request in the shortest time or for the less cost possible. Reliable resource selection can also be a point of view, where less error can occur, or a secure one that ensures the safety of the job. Providers may expect from the broker to utilize more or less a specific resource, or gain as much as they could from the resource utilization. An alternative method is to serve the user requests as to keep the balance of the load on the grid.

4. Survey of the Resource Brokers

The properties of the taxonomy were gathered from 14 Grid Brokers. Table 1. shows the examined brokers, and gives a short description of their architecture and operation. The columns correspond with the groups of categories described in section 3. This survey displays the main properties of the brokers. It indicates how the categories of the taxonomy are implemented and used in different solutions.

Table 1. Survey of Grid Brokers

Grid Broker	**Middleware Support**	**Job handling**	**Scheduling**
AliEn RB [11]	Alice	File transfer optimization, fault tolerance by multithreading	Push and pull task assignment
Apples [12]	GT 2	Parameter study support, event-driven rescheduling	Centralized adaptive scheduling with heuristics, self-scheduled workqueues
EZ-GRID Broker [13]	GT 2, 3	GUI for job handling, transparent file transfer	Own information service with dynamic and historical data, Policy Engine Framework for provider policies
GRIDBUS Grid Service Broker [14]	GT, Unicore, Alchemi	Failure management and application recovery, parameter study, API support (XPML description file)	Economy-based and data-aware scheduling
GridWay [15]	GT	Job migration support (checkpointing, resubmission), API support	Decentralized (or centralized) scheduler, adaptive scheduling
GRIP Broker [16]	GT 2, 3, Unicore	Ontology Engine for translating different job description	Ontology Engine for translating different information service data
GRUBER [10]	GT 3, 4	SLA-based resource sharing in multi-VO environment, disk qouta considerations	Internal site monitoring feature, various user-oriented policies
GTbroker [17]	GT 2, 3	Periodic and event-driven rescheduling, automated file staging	User-oriented policies, additional dynamic information
JSS RB [18]	GT 4, NorduGrid ARC	WS-Agreement for advance reservations, resource filtering by user requirements, file staging and replication support	Scheduling algorithms with benchmark-based execution time and transfer time estimation
KOALA [19]	GT 2, 3	Periodic and event-driven rescheduling, parallel co-allocated job handling, automated file staging	Processor and data co-allocation, own information service, hierarchical scheduling with queues, incremental claming policy

LCG-2 /gLite Broker [6]	LCG-2/gLite	Periodic and event-driven rescheduling, interactive job support	Eager or lazy policies, push and pull models for task assignments, provider-oriented policy support
NIMROD/G [20]	GT 2, Legion	Application level accounting, parameter study support, periodic rescheduling (Nimrod/G plan file)	Deadline and budget-based constrained scheduling, hierarchical and decentralized agent-based scheduler
OGSI Broker [21]	GT 3	User defined ranking in resource selection	User-oriented and provider-oriented resource owner policies, internal agent-based information system

5. Open issues and future work

From the survey and the taxonomy we can clearly identify, which properties are rarely used and which ones are highly supported. Regarding the whole taxonomy we saw that the Globus Toolkit is used by most of the brokers, therefore the RSL language is still the most widespread. The command-line interface is usual, and most of the brokers use a central scheduling architecture with just-in-time matchmaking optimized for minimal completion time. Rescheduling is widely used for fault tolerance.

On the contrary, the JSDL, which is a uniform standardized language, is rarely supported, yet. APIs, co-allocation, advance reservation and interactive job support should be provided by more brokers. A decentralized architecture could be a better solution in several cases, and own information systems should be built to gather more dynamic data and perform prediction-based matchmaking. As grids are heading towards the markets, provider-oriented policies should be more supported, and economy-based scheduling need to be considered. This solution requires QoS, so agreements must be supported by future brokers. To enhance reliability checkpointing and job migration should be targeted by future developments. Finally the most important thing to do is to provide all these broker properties to the users, making available more services, more middlewares and more resources in a transparent way. Interoperability is the key to achieve this vision.

Utilizing the existing, widely used and reliable resource brokers and managing interoperability among them could be new point of view in resource management. Our future work aims at developing a Meta-Broker that enables the users to access resources of different grids through their own brokers. Designing such an interoperable Meta-Broker, the following guidelines are essential: As standards play an important role of today's grid development, the interfaces must provide standard access. The architecture must be ‘plug-in based’ – the components should be easily extended by all means. The properties of the underlying components are also important; we need to be aware of the recent Grid Resource Brokers. The presented taxonomy and survey will provide the necessary information: how to interact with these brokers and which broker would provide the best service for a user task.

6. Conclusions

The presented taxonomy helps identifying and categorizing the most important properties of Grid Resource Brokers in various grid environments. We revealed the interfaces and relevant functionalities of the currently used brokers, which can enhance better user utilization and future development. With the presented survey users and scientists can have a better understanding of the operation and utilization of the current brokers. Developers should target issues that missing or rarely used in these solutions, but there is a definite need for them – to achieve this, the properties of the taxonomy give the guidelines. Furthermore, mapping the user needs and these broker categories, meta-brokers can solve the interoperability among Virtual Organizations and grids, which will be the main issue of future generation grids.

7. References

[1] I. Foster, C. Kesselman, “Computational Grids, The Grid: Blueprint for a New Computing Infrastructure”, Morgan Kaufmann, 1998. pp. 15-52.

[2] E. Afgan, “Role of the Resource Broker in the Grid”, Proceedings of the 42nd annual Southeast regional conference, 2004.

[3] K. Krauter, R. Buyya, M. Maheswaran, “A taxonomy and survey of grid resource management systems for distributed computing”, Software: Practice and Experience. vol. 32, 2, 2002, pp. 135-164.

[4] J. Yu, R. Buyya, “A taxonomy of scientific workflow systems for grid computing”, SIGMOD Rec., ACM Press, 2005, pp. 44-49.

[5] Globus Toolkit Homepage: http://www.globus.org/toolkit/

[6] EGEE Project Homepage: http://www.eu-egee.org/

[7] D. W. Erwin and D. F. Snelling., “UNICORE: A Grid Computing Environment”, In Lecture Notes in Computer Science, volume 2150, Springer, 2001, pp. 825-834.

[8] O.Smirnova et al., "The NorduGrid Architecture And Middleware for Scientific Applications", Springer-Verlag, LNCS 2657, 2003.

[9] A. Andrieux, K. Czajkowski, A. Dan, K. Keahey, J. Pruyne, J. Rofrano, S. Tuecke, and M. Xu, “Web Services Agreement Specification” (WS-Agreement), Internet, 2004, https://forge.gridforum.org/projects/graapwg/document/WS-AgreementSpecification/

[10] C. Dumitrescu, I. Foster, “GRUBER: A Grid Resource Usage SLA Broker”, 11th International Euro-Par Conference, LNCS 3648, 2005, pp. 465-474

[11] P. Saiz, A. Buncic, J. Peters, “AliEn Resource Brokers”, Conference for Computing in High-Energy and Nuclear Physics (CHEP 03), 2003.

[12] H. Casanova, G. Obertelli, F. Berman, R. Wolski, "The AppLeS parameter sweep template: user-level middleware for the grid", In Proceedings of the 2000 ACM/IEEE Conference on Supercomputing, IEEE Computer Society.

[13] B. Sundaram and B. M. Chapman, “XML-Based Policy Engine Framework for Usage Policy Management in Grids”, In Proceedings of the Third international Workshop on Grid Computing, LNCS vol. 2536, Springer-Verlag, 2002, pp. 194-198.

[14] S. Venugopal, R. Buyya, L. Winton, "A Grid Service Broker for Scheduling e-Science Applications on Global Data Grids", Concurrency and Computation: Practice and Experience, Volume 18, Issue 6, 2006, pp. 685-699

[15] E. Huedo, R. S. Montero, I. M. Llorente, “A framework for adaptive execution in grids”, Software: Practice and Experience. vol. 34, 7, 2004, pp. 631-651.

[16] J. Brooke, D. Fellows, K. Garwood, C. Goble, "Semantic matching of Grid resource descriptions", UoM. 2nd European Across-Grids Conference (AxGrids 2004), 2004.

[17] A. Kertész, “Brokering solutions for Grid middlewares”, in Pre-proc. of 1st Doctoral Workshop on Mathematical and Engineering Methods in Computer Science, 2005.

[18] E. Elmroth and J. Tordsson, “An Interoperable Standards-based Grid Resource Broker and Job Submission Service”, First IEEE Conference on e-Science and Grid Computing, IEEE Computer Society Press, 2005, pp. 212-220.

[19] H.H. Mohamed, D.H.J. Epema, "The Design and Implementation of the KOALA Co-Allocating Grid Scheduler", European Grid Conference, Amsterdam, 2005.

[20] R. Buyya, D. Abramson, J. Giddy, “Nimrod/G: An Architecture for a Resource Management and Scheduling System in a Global Computational Grid”, The 4th International Conference on High Performance Computing in Asia-Pacific Region (HPC Asia 2000), IEEE Computer Society Press, 2000.

[21] Y. Kim, J. Yu, J. Hahm, J. Kim, et al., “Design and Implementation of an OGSI-Compliant Grid Broker Service”, Proc. of CCGrid, 2004.