Grid Resource Discovery Based on Centralized and Hierarchical Architectures

Deniz Cokuslu1,2,3, Abdelkader Hameurlain2, Kayhan Erciyes3
1Izmir Institute of Technology, Department of Computer Engineering, Turkey
2IRIT, Paul Sabatier University, France
3Izmir University, Turkey

Abstract

Grid systems have gained remarkable importance in the last decade since resource requirements of recent applications increased drastically. Resource discovery in grid systems is a fundamental task which provides searching and locating necessary resources for given processes. There are many different approaches in literature for this problem. Grid resource discovery using centralized and hierarchical systems is a widely used approach which has resulted in many tools to become de facto standards of today’s grid resource management. In this paper, we propose a survey of recent grid resource discovery studies based on centralized and hierarchical systems. We provide synthesis, analysis and evaluation of these studies by classification. We also give a comparative study of different classes proposed.

1. Introduction

Grid system is a large scale distributed environment which provides a high number of powerful resources to its users [1]. It differs from classical distributed systems by their heterogeneous and dynamic nature. While resources are nearly identical and stable in a classical distributed system, in grid systems resources are highly heterogeneous and dynamic in terms of both stability and dynamicity of their properties. Grid systems provide many types of resources such as computing resources (CPU cycles, storage, network bandwidth, memory etc.) and services (access to specific data, shared software etc.). Web services in this area are very popular software entities that offer an easy and standard way to access functionalities of different kinds of platforms [2]. In this sense, web services can be used in designing useful Grid management tools. To realize this, many standards such as UDDI [3], WSDL [4] and SOAP [5] are developed and already in use. Since grid services are very similar to web services, their convergence leads researchers to develop grid services as abstract sub classes of web services. The mapping between them is used in many grid management problems such as resource discovery which is one of the most common problems in both domains. Today most of the grid management tools are based on web services. Globus Toolkit [6], Alchemi [7], Unicore [8] and gLite [9] are the most popular examples of such systems. Especially Globus Toolkit (GT) is broadly accepted to be the current de facto standard in grids. Globus provides many grid services such as resource discovery, allocation and monitoring by extending standard service interfaces [10].

Resource discovery problem in grid systems can be defined as searching and locating resource candidates which are suitable for executing jobs in a reasonable time in spite of the dynamicity and large scale of the environment. Success of Grid systems mainly relies on efficient usage of the right resources. Resource discovery is an important step in finding these resources. But the characteristics of the grid systems make the resource discovery a time consuming process which can decrease the performance of the whole system. Several methods have been proposed to solve the resource discovery problem in Grid systems. They are classified into three main categories in the literature [11]: methods based on centralized and hierarchical systems, peer to peer (P2P) based systems, and agent based systems. Centralized and hierarchical systems emerged as suitable approaches which provide easy to access tools for Grid services [12-17]. In such systems, resource information is stored and updated in central or hierarchically located servers, and resource discovery is realized by querying these servers. Their drawback is that they do not scale well in grid systems in which frequent updates and large numbers of requests exist. On the other hand, agent based systems [18-22] are attractive in grid systems because of their autonomy property. They have capabilities to determine new migration sites according to their migration policies for the distribution of resource discovery queries. Peer to peer systems [23-26] are also suitable for grid services because of their scalability and reliability properties. Most recent P2P techniques use structured distributed hash tables (DHT) to manage resource information which increase the performance.
of the resource discovery drastically. Moreover, they ensure the reliability since all nodes in the system involves the resource discovery process. A comprehensive study of survey about P2P based and agent based grid resource discovery can be found in [27].

Web service based grid management tools use mostly centralized and hierarchical resource discovery mechanisms. There are only few web services based tools which use other classes of approaches such as peer to peer resource discovery [28-29]. In this paper, we focus on centralized and hierarchical systems and present a survey of recent studies which provide grid resource discovery services using centralized and hierarchical approaches [8, 10, 13-17, 30-34]. We classify them according to the topology of resource information. In centralized methods, resource information is stored in a central server, whereas in hierarchical methods resource information is divided and partially distributed to several locations. We give detailed analysis for each study. We also evaluate them according to important qualitative criteria namely, scalability, dynamicity, reliability and support for multi-attribute, dynamic-attribute and range queries. We believe that those criteria are very important in analyzing grid algorithms. More precisely, scalability is a very important measure since grids are large scale environments in their nature. Dynamicity is another important measure because nodes in grid systems might be highly dynamic in terms of joining and leaving the system, mostly without any notice. Reliability should also be considered because in some cases erroneous query results may cause irrecoverable faults. And lastly, support for different types of queries is a decisive criterion in selecting the methodology in most cases. The main contribution of this paper is the detailed analysis of examined studies within a specified classification, and comparison between classes. The rest of this paper is organized as follows: Section 2 gives analysis and evaluation of grid resource discovery methods based on centralized systems. Section 3 analyzes and evaluates grid resource discovery methods based on hierarchical systems. Section 4 compares two classes of resource discovery methods. Finally, in Section 5, we conclude the paper by giving an overall summary and remarks about examined methods and express our future plans.

2. Grid Resource Discovery Using Centralized Systems

Grid resource discovery using centralized systems is excessively studied and implemented in grid environments. Most of those studies profit from existing information provider systems such as LDAP (Lightweight Directory Access Protocol) or UDDI (Universal Description Discovery and Integration). In this section, we examine some recent studies in this class, and evaluate them according to the selected evaluation criteria.

2.1. Synthesis and Analysis

In [17], Yu et al. developed a web services based grid service publication directory system (GMD). The system provides service for resources and clients via web by using XML formatted messages. The GMD consists of two main components. The first component, GMD Portal Manager, provides a web based platform to manage registration and publication of resources. A user which would like to share its resources uses the portal in order to advertise its resources to the central repository. The updates of resource information are accomplished by the resources’ owners. Therefore, to compensate the dynamicity of grid, users need to update resource information either manually or with the help of a third party utility. The second component, GMD Query Web Service is a query processing web service which is used to discover required resources that meet the query requirements. GMD is a centralized grid resource management system. It uses a central resource repository database and central query processing engine. Because of its centralized nature, the system may suffer from bottleneck and single point of failure problems in a large scale grid system. Moreover, it is not suitable for dynamic grid environments since the dynamicity requires manual updates. On the other hand, the system supports multi-attribute and range queries since the repository is a database system in which those types of queries can easily be supported. But it does not support dynamic-attribute queries. Kaur and Sengupta presented a centralized resource discovery mechanism for grids which relies on web services [14]. Their study consists of four main components. The UDDI rich query model uses UDDI standard [35] to discover grid services. The standard is used to hold grid resource information as key-value pairs in the UDDI database. The second component GWSDL (Grid Web Services Description Language) is used to describe grid services in an extended version of Web Services Description Language. GWSDL has OGSI (Open Grid Services Infrastructure) support and plays an important role on standardizing the communication between OGSI and web services. The third component is SOAP which accomplishes communication between web services in the grid environment. The last component is HTTP which provides easy to use interface to post requests and get responses. The web service processes requests by querying UDDI database. The system uses central servers and databases to run the web service. Therefore, it contains bottlenecks and single point of failures. Moreover, since the resource information is
held in a repository database, dynamic attribute queries are not supported. On the other hand; the proposed web service supports range and multi-attribute queries. In [15], Molto et al. proposed a metascheduler grid service that can be accessed through the network by users who are interested in task allocation and scheduling in computational grids. Metascheduling is the process which takes a job demand, allocates suitable resources, runs the job and returns the result to the user. The proposed system is a service oriented architecture which can be used via the web without any platform requirements. It is developed on top of Globus Toolkit which uses Metacomputing Directory Service (MDS) for the resource discovery. The proposed system in this study fills the gap between a simple user interface and complex Globus resource management primitives. In terms of resource allocation, the proposed system additionally implements a transparent caching component which stores static resource information, which is assumed to be unchanged, for the future queries. The resource discovery phase is held by first interrogating a GIIS (Grid Index Information Service) or a BDII (Berkeley Database Information Index) service. The proposed system uses standard protocols like SOAP and XML for data exchange and the whole process is held by a central server. Therefore, the central server may easily become a bottleneck and single point of failure in highly dynamic and large scale grid environments. Moreover, since dynamic attributes are updated in the database in discrete intervals, dynamic-attribute queries are not supported. On the other hand the system supports multi-attribute and range queries. In [8], authors present a web services based grid middleware (UNICORE 6) in which resources are represented by web services. UNICORE is based on open standards such as Web Services Addressing (WS-A) and Web Services Resource Framework (WS-RF), thus it is compliant to many emerging grid standards such as OGSA (Open Grid Resource Architecture). In UNICORE, each resource is defined by standardized forms provided by WS-RF (Web Services Resource Framework) and is stored in central locations by using UUDB (UNICORE User Database). The services provided by UNICORE consist of five stateful web services that allow job submission, easy access to storage resources, and file transfers. The first service, TSS (Target System Service), manages the resource information in grid. Jobs are described by standardized job submission description language (JSDL). TSS analyses submitted jobs and searches for available resources. In [31], authors proposed a Meta-computing Directory Service (MDS) for resource management in grid systems. Proposed study is based on LDAP in which metadata is maintained by central servers. In MDS, resources are represented by MDS entries in the LDAP server which are specialized data structures. Resource information maintenance and responses to the queries are handled by LDAP servers. The initial MDS design is evolved and enhanced (MDS-2) to work in a hierarchical manner in [30]. MDS-2 provides a configurable information provider component called grid resource information service (GRIS) and a directory component named grid index information service (GIIS). In MDS-2, dynamic and static resource information is provided by the GRIS service. Provided resource information is stored in aggregate directories using GIIS. An information provider registers itself to a directory regarding the local and virtual organization specific policies. The provider then updates its resource status in its registered directory. If a resource provider does not update its resource information for a predefined period, the directory assumes that the provider has become unavailable and deletes the provider from its directory. As long as a provider exists in a directory, it is included in results for relevant discovery queries.

2.2. Evaluation

The grid resource discovery using centralized systems provide grid middleware developers an easy to use interface to manage grid resources. They keep grid resource information by using centralized databases. In a large scale grid environment, the centralization of the service may easily create bottlenecks on the central servers. The bottleneck problem may arise both because of frequent resource updates or large number of query requests waiting to be processed. The centralization causes another important problem in dynamic grids as being a single point of failure. Failure of one of the central servers in the system may cause the whole system to become unavailable. In some approaches, the idea of replication of central servers is depicted in order to eliminate single point of failures. But replication of servers in a large scale dynamic grid may be very expensive in terms of communication costs. The proposed systems support the multi-attribute and range queries since the resource information is stored in databases which are capable of processing complex queries. But since the update of dynamic resource attributes are held in discrete intervals, most of these systems do not support dynamic-attribute queries.

3. Grid Resource Discovery Using Hierarchical Systems

In recent years, as the size of the grid environments grows, researchers directed their attention to hierarchical systems to overcome the problems caused by centralized systems in resource
discovery. In this section, we analyze and evaluate some recent grid resource discovery methods which use hierarchical systems.

### 3.1. Synthesis and Analysis

In [13], Elmroth and Tordsson proposed a grid resource broker and job submission system. Their algorithm is a complete grid management middleware. In their design, they take advantages of many existing tools by utilizing and extending them. For the resource discovery phase, they benefit from Grid Laboratory Uniform Environment (GLUE) project which provides information related to the resources. The server side module which is called job submission module is composed of seven components. The first component, in which we are interested in, is the InformationFinder component. The InformationFinder component discovers grid resources and provides information about them. InformationFinder component works in cooperation with index servers which are responsible for indexing resource information. The index servers are organized in a hierarchical topology. Once a query is received, the InformationFinder performs resource discovery by querying each index server. Then, the resources which are found to be suitable for the query are contacted to get more detailed up-to-date resource information. Both static and dynamic information about the resources are retrieved in this step. The InformationFinder also retrieves usage policies, allowing it to discard resources where the user is not authorized to submit jobs. Then the extracted resource information is passed to the higher level components of the system to accomplish job submission processes. The system stores the resource information in information servers which are organized in a hierarchical structure. The hierarchical organization reduces the probability of bottleneck problem. But it cannot solve single point of failure since failure of a server in the hierarchical organization may result in a large part of the resources to be excluded from the queries. On the other hand, since the resource information is stored without any hash function and since it is verified by contacting the candidates directly, the system supports all multi-attribute, range and dynamic-attribute queries. In [16], Ramos et al. proposed a web service for resource discovery in grids, based on Globus Toolkit (GT3). They proposed a hierarchical topology in which the grid environment is divided into virtual organizations (VO). In each VO, there are master and slave nodes. The master nodes are responsible for updating the resource database and the slave nodes are responsible for retrieving resource information from each machine which composes the grid system. The resource discovery is realized by the preparation of a configuration file which includes the requested resource information. Upon detecting the file, the resource discovery mechanism generates an XML file which is going to be manipulated by resource discovery service. After this new configuration is distributed to all slave nodes, the service is restarted in all nodes. By doing this, a new service containing the new search is deployed. When a slave machine receives such a query, it checks if the requested resources are satisfied and returns results to its master node. Then the resource discovery service on the master machine processes the retrieved files and returns the results to the requester. Since the proposed system is a hierarchical web service based system, it decreases the possibility of bottleneck problem. On the other hand; since the resource discovery task is managed by master machines, failure of a master machine may result in a large number of slave machines becoming invisible to the system. This system supports all multi-attribute, range and dynamic-attribute queries since the resource discovery is held by resources on-the-fly without any hashing function. Beheshti and Moshkenani proposed a resource discovery method by joining agents, ontologies and web services [32]. They used SOA concept in which the discovery service is placed on different platforms. Information about resources are sent to coordinating services and published in a portal which is accessible by the users. The proposed system uses XML based semantic information related to resources. Web services are used to update the registered resource information in the system. They provide dynamic resource list for the users and also make them accessible. The metadata related to resources is distributed to some nodes in which the information is updated by the use of web services. Updates are handled by agents in resources which provide all the time up-to-date resource information. In [33] Li et al. presented a grid resource discovery model which is inspired by network routing mechanisms. In their model, there are three types of nodes, namely, resource nodes, router nodes and requesters. Each resource node is bound with a router node and posts its resource information to its router periodically. Every router is connected to at least another router node. Router nodes store resource information in routing tables and share their routing tables by communicating their neighboring routers. When a router receives a request from a requester, it checks routing tables and chooses a neighbor to transfers the request. A request may be transferred by several routers until it arrives at a provider satisfying the requirement. The model uses TTL (time-to-live) parameters to limit the dissemination range of requests. For the routing of requests, they proposed an efficient routing algorithm which is called SD-RT (Shortest Distance Routing-Transferring) which chooses a route which is nearest to a router. This model considers both dynamic and multi attribute queries but the periodical update of the resource information limits...
the utility of dynamic attribute queries. Moreover, usage of TTL parameters may cause false-positive errors. In [34], Yin et al. proposed a 3-layer hierarchical grid resource discovery method. The first layer is composed of resource nodes. The second layer, resource information layer, consists of the nodes which store resource information. Each node in the first layer is linked to a resource information node in the second layer. The nodes in the resource information layer form virtual organizations and each virtual organization has a super node. Super nodes form the third layer, the index information layer. The size of each virtual organization in the resource information layer is restricted to a predefined value in order to ensure scalability. Resource nodes send their resource information to the resource information layer periodically. When a resource request is submitted, local area in the resource information layer is examined first. If the resource cannot be found, the request is forwarded to the index information layer. When the requested resource is found, before returning to the user, the resource node is checked to find whether it still meets the requirements. If it does, the contact information of the resource node is returned to the requester. If not, search phase continues until a resource is found. The algorithm ensures scalability by limiting the size of the virtual organizations. But the super nodes are assumed to be stable which makes the algorithm unreliable for grid systems. Multi attribute, range and dynamic attribute queries are supported.

3.2. Evaluation

The hierarchical systems based grid resource discovery algorithms provide a more scalable platform than the centralized ones and still provide a simple user interface to manage grid resources. In a large scale grid environment, the hierarchical topology of the service decreases the probability of bottleneck problem. But single point of failure problem still exists since failure of one of the master servers in the system may cause a large part of the nodes become invisible to the queries. All proposed algorithms in this class support the multi-attribute and range queries since the resource information is stored in databases which are capable of processing complex queries. And since, in many studies, the information is verified in the resource nodes after querying databases, the proposed algorithms also support dynamic-attribute queries.

4. Comparison

The summary of comparison between resource discovery using centralized and hierarchical systems can be seen in Figure 1. Both centralized and hierarchical methods behave closely against nearly all the evaluation criterions. The main difference is in scalability and reliability. The centralized systems suffer from the bottleneck problems in large scale. There also exists the single point of failure problem. Even though some studies propose to replicate the centralized index server, this procedure might be very expensive in terms of messaging complexity in large scale. On the other hand, hierarchical systems distribute the load into many locations instead of one central server. This property increases the scalability of the system by distributing load on index servers. They also decrease the effect of single point of failures. In case of a failure of an index server, a part of the system becomes unreachable instead of the whole. Support for dynamic attribute queries requires that the query is processed within the resource nodes. Since the idea in the examined solutions is indexing the resource information in central locations, they do not support dynamic attribute queries. Some of the examined algorithms propose solutions for this problem, but the solutions are independent from the classification that we propose.

![Figure 1. Comparison summary between two resource discovery methods](image_url)

By taking the evaluations into consideration, we can say that the centralized methods are not suitable for the large scale environments. But they might be well suited to the systems in which the scale is small and indexing server is reliable. In such cases centralized systems can be used effectively. On the other hand, hierarchical methods are more suitable
for environments in which scale is bigger since the load is distributed to many locations. But even the load is hierarchically distributed; those methods may still suffer from bottleneck problem in large scale.

5. Conclusion and Future Works

In this paper, we have synthesized and analyzed some recent grid resource discovery methods which are based on centralized and hierarchical systems. We evaluated them by defining some qualitative criteria, and compared different classes of methods with each other. These types of resource discovery algorithms seem to have many disadvantages in large scale dynamic grid environments. But their simple design brings them to the foreground and makes them suitable techniques to be used in many grid management tools. Most web service based grid management tools use centralized and hierarchical grid resource discovery techniques. With regards to the examined methods, we can say that centralized and hierarchical systems based approaches are suitable for small scale grid environments in which the dynamicity of nodes is low. In such environments, they provide a very simple and standard resource management platform. There is vast amount of ongoing research in this topic which is aimed at solving problems mostly caused by scalability issues. P2P based [23-25] and Agent based [18-20] approaches in grid resource discovery are two of the most promising approaches in this field. A very comprehensive and detailed survey about those grid resource discovery approaches can be found in [6, 26-27].

We believe that combining advantages of different RD approaches would result in useful studies for resource discovery in grids. For example, by using web services, one can provide a simple interface to the users, while for the RD module, by using mobile agents or P2P techniques, a scalable, reliable and also easy to use resource discovery middleware can be designed. To the best of our knowledge, there is no research work which is directly focused on this subject. For this purpose, we are planning to propose a new cross-layer design approach in resource discovery which combines topology control, efficient multicasting and web services together. Firstly, we are planning to build virtual clusters in the grid environment which are represented by their clusterheads. Then we are planning to build an efficient topology, such as a spanning tree, between clusterheads in order to ease routing of the messages between peers. After building the topology, we are planning to deploy an efficient routing algorithm which uses all the advantages of the proposed topology. Finally we are planning to join web services in order to enable integration of our study to the current grid middleware systems.

6. References


