Military Technical College
Kobry El-Kobbah,
Cairo, Egypt

8th International Conference
on Electrical Engineering
ICEENG 2012

A survey of resource discovery in computational grids

By

Hesham A. Ali* Alaa E. Abdel-Hakim** Mohammed A. Ahmed***

Abstract:

Grid computing is a wide spread technology in recent years. It offers an effective way to build high-performance or high-throughput computing systems, allowing users to efficiently access and integrate geographically distributed computers, data, and applications. Searching and locating the resource which match the user's requirements in an efficient and timely manner is the important phase in grid computing which called resource discovery. Discovering resources in grid environment is complex due to the heterogeneous nature, dynamic availability of resources, resources are owned by different individuals and organizations and each having their own resource management policies i.e. different access and cost models. There are many different approaches in literature for solving this problem (e.g. Centralized-based, Hierarchical-based, Agent-based and P2P-based). This paper provides a survey and analysis on ongoing researches as well as evaluation summery of those approaches on this specific area. We believe that this survey would be useful for academic and industry based researchers who are engaged in the design of scalable computational Grid.

Keywords:

grid computing, resource discovery, hierarchical, p2p, agents.

* Computer Engineering Department, Mansoura University, Egypt
** Electrical Engineering Department, Assuit University, Egypt
*** Electrical Engineering Department, Assuit University, Egypt

1. Introduction:

Grid is a collection of shared, geographically distributed hardware and software that allow user to solve large-scale problems [1] and enable sharing, selection and aggregation of suitable computational and data resources for solving large-scale data intensive problems in science, engineering, and commerce. It differs from classical distributed system by their heterogeneity, dynamic nature and huge large scale. The Grid environments comprise heterogeneous resources (PCs, workstations, clusters, and supercomputers) with computational resources (CPU, input/output (I/O), memory and/or network bandwidth), services (access to specific data, shared software etc.), and applications (scientific, engineering, and commercial) and users: producers (also called resource owners); consumers (also called end-users) have different goals, objectives, strategies, and demand patterns.

Heterogeneity of resources mean that there are very variation in types of resources as mentioned above or mean that resources themselves have different values for their attribute (e.g. CPU may be INTEL or AMD or SPARC etc. ,and operating system of machine may be WINDOWS or LINUX or MAC etc. and so on).

From dynamicity view point we can see that the availability and status of recourses within each organization change dynamically over time, and resource can be join or leave or may be fail in unpredictable way. Table (1) Summaries different resource classes [2] which belong to Grid.

From Large scale view point we mean: (i) high numbers of data sources, users, and computing resources which are heterogeneous and autonomous, (ii) the network bandwidth presents, in average, a low bandwidth and strong latency, and (iii) huge of data volume membership protocol defines the multi grouping approaches for the management of different groups or organizations.

Table (1): *Resource Classes Examples*

Resource class	Description
Computational resources	Computing capabilities provided by computers, supercomputers, workstations such as CPU, memory, network and I/O bandwidth etc.
Storage resources	Storage space such as disks, external memory, etc.
Device resources	Specific devices such as instruments, sensors, etc.
Software resources	Operating systems, software packages, Web services, etc.
Data resources	Various kinds of data stored in file systems, xml files or databases.

Resource management is the process of managing available resources and system workload accordingly. Resource management scenarios [3] often include resource discovery, resource scheduling, resource monitoring, resource inventories, resource provisioning, fault isolation and variety of autonomic capabilities and service level management activities.

Resource Discovery (RD) is the process which include resources description, resources advertisement, resources organization, management of resources attributes and characteristic (configuration, availability, usage policy and constraints), resource lookup and locate resource which satisfactory the user's request. Users are not interested in where resources actually are, just by given a description about resources they desired, the RD mechanism will find set that match user's description if their exist one. For this reason, RD is a vital part of a Grid system, and an efficient RD infrastructure is crucial to make the distributed resource information available to users in a timely and reliable manner. So, ineffective and inefficient RD mechanism in grid environments affect the overall system performance. Discovering a specific resource in traditional computing systems is relatively easier than in grid computing because the number of shared recourses is small and all recourses are under central control[4]. However, resource discovery in large-scale grids is very challenging process due to the potential large number of resources, and their heterogeneity , distributed ownership, and dynamic nature.

The remainder of this paper is structured as follow: Section 2 discussed related work of survey of resource discovery in Grid. Overview of the grid RD system is presented in section 3. In section 4 we discussed the taxonomy approaches of RD. The survey of recent existing researches on RD and provide analysis' summery for each is given in section 5. section 6 present the evaluation summery of RD approaches. The conclusion in section 7 is presented.

2. Related Work:

There have some studies on survey and taxonomy of resource discovery in grid. The most famous ones is done by P.Trunfio et al. [5] and A. Hameurlain et al. [6],[7]. In [5] authors review the most promising grid system that use p2p technique for resource discovery until 2006. The work [6] present ongoing researches until 2009, authors surveys Grid RD studies based on centralized and hierarchical approaches only. In [7] authors provides a survey and a qualitative comparison of the P2P-based and agent-based approaches for RD in grid until 2008. In [8] author cover the efforts and algorithms for RD based on p2p approach from 1997 to 2003. In [9] authors focus on agent-based approach and they cover some of efforts that have been presented in p2p until 2008. Semantic approach is surveyed by [10]. As we see that the most recent work which surveys efforts and algorithm of grid RD approaches was in 2009. However a lot

of fresh studies have been done after that. This survey covers RD in general, provides a taxonomy to differentiate between various approaches of RD in Grid, as well as cover and analysis the most recent of efforts that each approach is included. Moreover, evaluation summery for all approaches and methods in this specific area will be presented.

3. Overview on Grid RD:

3.1. Components of Grid RD

Grid RD consist of four main components namely *Description*, *Dissemination*, *Indexing* and *Discovery* (which is composed of *search* and *selection*) (*See* figure 1) .

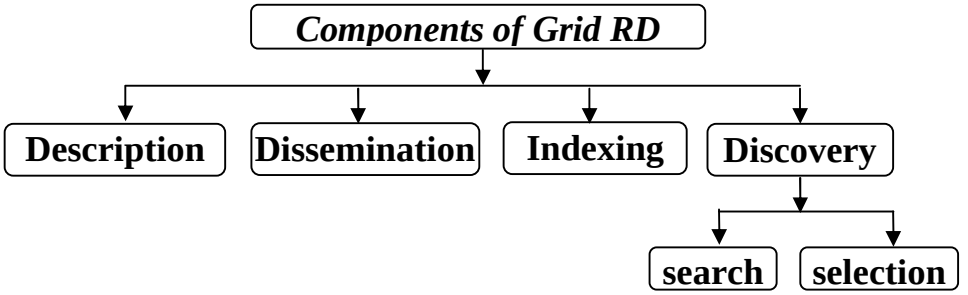

Figure (1): *Components of Grid RD*

3.1.1. Description:

Description In general; each resource in grid system will describe with set of attributes corresponding to its characteristics and status. Those attributes are either static or dynamic. Table 2 summarizes these types.

Table (2): *Resource Classes Examples*

Type	Definition
Static attributes	Refer to resource characteristics that do not change frequently such as “CPU architecture”, “CPU speed”, “Operating System name”, “Physical memory size” , “Software installed”, “Secondary memory”, “Network bandwidth” and “ Network location”.
Dynamic attributes	Are associated to fast changing characteristics over time such as “CPU load ”, “Free Memory”, “Queue length” and “Network Bandwidth utilization ” etc.

There are two ways to describe the resources and requests in Grid RD system on the basis of Symmetric or Semantics as shown in figure 2. And these ways as follow:

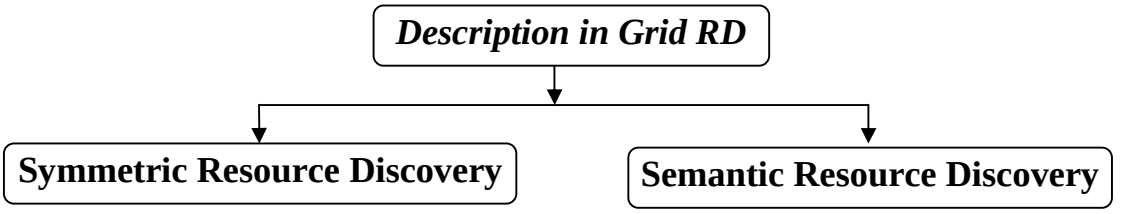

Figure (2): *Description in Grid RD Taxonomy*

3.1.1.1. Symmetric-Based Resource Discovery:

Symmetric approach is the traditional RD methods to describe and search for specific resource and its common called keyword-based matching, since values of attributes advertised by resources are compared with those required by user' request. For the comparison to be meaningful and effective, the resource providers and consumers have to agree upon attribute names and values. Thus the percentage of successful Resource Lookup Quires (RLQs) [2] are affected significantly by the increase in query rate under average queue size. Moreover, most centralized and decentralized RD mechanisms are based on syntactical approach, where there is high possibility to miss relatively close resources [11].

3.1.1.2. Semantic-Based Resource Discovery:

Semantic technology [12] is type of information, data models and mechanism that are used to describe resources and job requests using an expressive Ontology language; instead of exact syntax (e.g., attribute-value pairs) matching. Ontology is defined as the formal specification of a vocabulary of concepts and axioms relating to them [13]. It formally specifies how to represent objects, concepts and other entities that are assumed to exist in some area of interest and the relationships among them [5]. Establishing relationships between domain concepts allows us to understand the concept not merely by its properties, but by its presence in relation to other concepts within the ontology. Ontologies that can be machine processable are created using semantic markup languages. These languages are developed under the Semantic Web [14].

Semantic similarity function is used to calculate the similarity between two concepts. Similarity between two concepts shows the association degree between them. Similarity function is defined as: $sim\ (x, y): C*C \rightarrow [0, 1]$ [15], the output of this function is a factual number in the span of [0, 1] which shows the extent of association degree between two concepts of x and y. If the output is zero, it means lack of association and if it's one, it proves full similarity between the two concepts.

3.1.2. Dissemination:

Dissemination is the mechanism which use to introduce newly joined resources to the grid environment [16]. This stage also includes periodically update process for the dynamic attributes for those resources. Figure (3) shows the taxonomy of resource dissemination [2]. In *Batch dissemination* each Grid machine batch up information of its resources and then this information periodically disseminated through the Grid. Information can be sent from the originating machine to other machines with two ways: (i) originating machine pushes information to other machine or (ii) another machine in the Grid request the information from the originating machine with pull mechanism. On the other hand, an *online* or *on demand* approach; information is disseminated from the originating machine immediately. In this case the information is pushed to other machines in the Grid.

Choosing an unsuitable resource dissemination algorithm may cause severe network traffic inside the computing environment also selecting the appropriate attributes of a resource for advertisement is also very important.

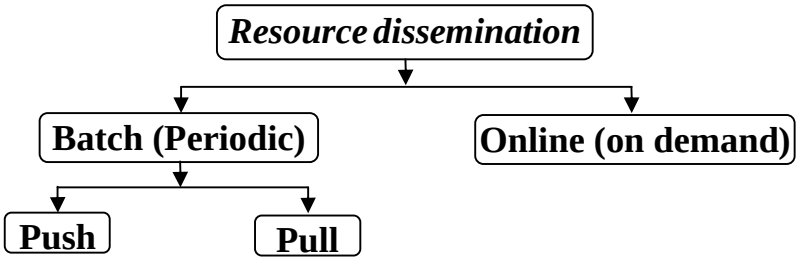

Figure (3): *Resource Dissemination Taxonomy*

3.1.3. Indexing:

An Index Service is a special purpose Grid Service that aggregates and indexes metadata related to the resources provided by the Grid hosts of a Grid organization. In other words, indexing means the registration or storing the resource information and how long to keep this information. It has two main aspects: registry architecture and update mechanism. The former refers to the registry location and its distance with regard to the resource providers in the network, whereas the latter is a monitoring scenario for the status of the registered advertised resource capabilities.

3.1.4. Discovery:

There are two steps for discovery phase are: (i) The mechanism for distribute a request from consumer node to node responsible for indexing the requested resource and this step called *search*; (ii) selecting a number of resources that have been discovered in

search step to perform the users tasks and this step called *selection* [17]. Taxonomy of RD in Grid will discussed in the next section with details (see Figure 4). Four types of queries or requests apply to each attribute involved in RD are "*Exact match*" , "*Partial match*" , "*Range match*" , "*Boolean match*". Table 3 summarizes these types.

It is noticed from mentioned overview that dissemination and discovery compliment each other [2]. The dissemination is initiated by the resource which needs to be discovered, whereas discovery is done by the application (e.g. user) to find a suitable resource.

Table (3): *Types of queries involved in RD*

Query	Definition
Exact match	Refer to the query which specify exact desired values for all resource attributes sought.
Partial match	Which specify values for a number of selected attributes.
Rang	Specify range values for all or some of the attributes.
Boolean	In which certain Boolean conditions are specified for all or some of the attributes.
multi-attribute	is composed of a set of sub-queries on single attributes, each sub-query fit in one of the four types from the above list and the involved attributes are either static or dynamic.

4. Taxonomy of RD in Grid:

Regardless of how resource and request are described, distribution of resource attributes and user requests (i.e. searching about specific resource) can be classified into many categories as shown in figure (4). Many researching efforts have been invested in designing RD techniques for the large-scale Grid systems.

Methods of Grid RD are mainly based on the Query and Agent-based techniques. In a query based discovery the resource information store is queried for resource availability, whereas in an agent-based discovery; agents traverse the Grid system to gather information about resource availability [18]. Thus, the basic difference among query-based and agent-based mechanisms [19] is that in agent-based the agent makes RD decisions based on its own logic, whereas in query-based systems; RD is done by the predefined logic.

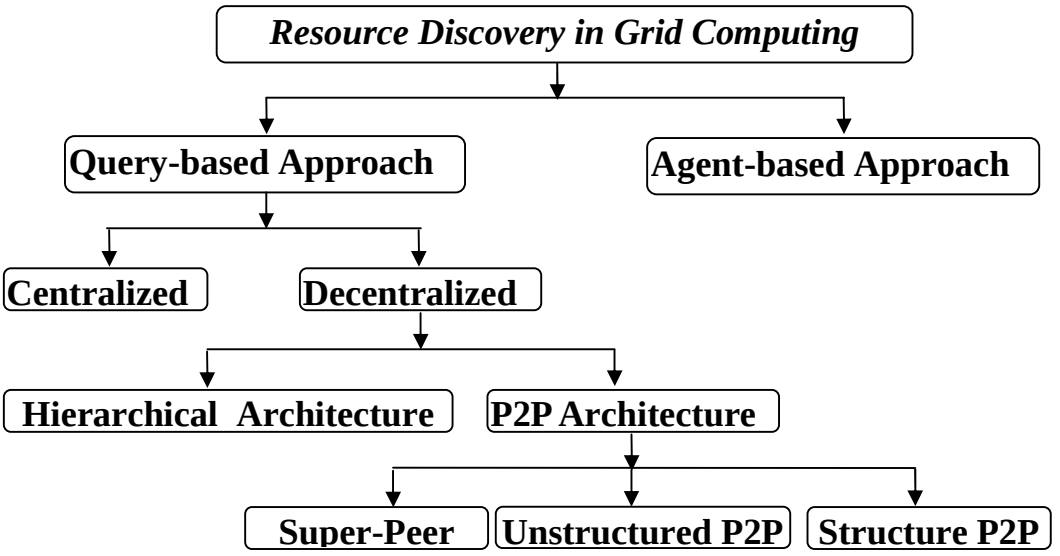

Figure (4): *Resource Discovery Taxonomy*

4.1. Agent-Based Resource Discovery:

Software agents are programs that act on behalf of people and can accomplish a task since their autonomous [20] property in which they have control both over their internal state and over their own behavior and acting independently of the supervision of the user. By providing the ability to transport themselves (i.e. Mobile Agents) between different systems, they can carry internal information which was obtained by each visited member. Mobile Agent (MA) is an intelligent agent with mobility, which can be moved independently from one host to another host on the network, and complete specific tasks such as searching, filtering and collecting information, even do commercial activities on behalf of users. MA has the autonomy, responsiveness, initiative, communication and mobility and other characteristics. Thus, Agents with those useful characteristics can be considered as an applicable idea in RD phase of Grid architecture [9].

4.2. Query-Based Resource Discovery:

In this category; user sends message with the needed requirements of resources to the node which contain information about those resources by discovery mechanisms, for example, network directory based systems mechanisms such as Globus MDS [21] use parameterized queries that are sent across the network to the nearest directory, which uses its query engine to execute the query against the database contents. Query-based is the most common used approach in researches and studies of Grid RD. Indexing resource attributes and searching (distribution of queries) may be happen with centralized or decentralized mechanism as details in remainder of this section.

4.2.1. Centralized approach:

In centralized approach, there is central database in which all grid resource information are kept. A user which would like to share its resources uses the portal in order to advertise its resources to the central repository. When users need specific resources for their tasks also send their request with description of requirements to the central directory, then directory server will response with the positive or negative result to that query. If required resource is found, the address of the node that host the resource will be returned to the user and then user will negotiate with the owner of that resource. for example MDS [21], also, Condor's Matchmaker[22] which adopts a centralized mechanism to match the advertisement between resource requesters and resource providers using Central Manager (CM). Centralized approach will discuss with detail in next section.

4.2.2. Decentralized approach:

Here, Grid resource information store in decentralize sites along grid infrastructure. Two main categories of this approach are Hierarchical and P2P method.

4.2.2.1. Hierarchical architecture:

As the size of the Grid system grows, researchers directed their attention to hierarchical systems to overcome the problems caused by centralized systems in resource discovery . Those problems will demonstrate in subsection 5.2.1. In these approach queries are processed hierarchically since servers have been organized hierarchically so that each server is responsible for partitions of resource information for example Globus MDS-2 [23] uses a Lightweight Directory Access Protocol (LDAP) based directory service for resource registration and lookup. Although this approach solves several problems of centralized approach, it still suffers from some issue as detail in subsection 5.2.2.1.

4.2.2.2. Peer-to-Peer architecture:

In a decentralized P2P system, hosts (i.e. node) are arranged in a P2P overlay network. Adjacent peers, i.e. peers that can directly communicate through an overlay P2P connection, can belong to the same Grid organization or to different organizations. Fortunately, Grids and P2P systems share several features and can profitably be integrated [24], bringing benefits in both fields and resulting in future convergence.

Two major differences between P2P systems and Grids, however, determine their different approaches towards resource discovery. First, P2P systems are typically designed to share files among peers. Differently, Grids deal with a set of different resources, ranging from files to computing resources; Second, the dynamism of P2P systems comes from both nodes and resources since Peers join and leave at any time,

and thus do the resources shared among them. In Grid environments, nodes connect to the network in a relatively more stable manner[37]. The dynamism of Grids mainly comes from the fast-changing statuses of resources. For example, the storage space and CPU load can change continuously over time. Due to those differences, the existing algorithms which use in RD for p2p system Needs to be modified [25] to fit with the nature of grid computing, accordingly, several RD techniques have been adapted to work efficiently in Grid environment. P2P techniques are the more recent approaches which use for solving the problem of resource discovery in Grids.

The P2P-based algorithms can be classified into three classes depending on how peers are organized: unstructured P2P systems, structured P2P systems and super-peer systems.

4.2.2.2.1. Super- Peer architecture:

A super-peer acts as a centralized node for a number of regular peers, while super-peers connect to each other to form a network that exploits the P2P mechanisms at a higher level. The super-peer model is naturally appropriate for Grids, as a large-scale Grid can be viewed as a network interconnecting small-scale, proprietary Grids, also referred to as Physical Organizations (POs). Within each PO, one or more nodes (those that have the largest capabilities) act as super-peers, while the other nodes use super-peers to access the Grid and forward discovery requests.

4.2.2.2.2. Unstructured P2P architecture:

Unstructured solutions do not have strict rules about the topology of the network nor about the location of resources. Resources can be located by means of flooding mechanisms, namely first by querying neighbors of the local node and then propagating these queries progressively throughout the network. Although this approach is more scalable than a centralized registry, it can cause a lot of additional traffic overhead. To leverage this problem, queries are usually given a maximum time-to-live (TTL). This TTL value limits the maximum number of times a query can be forwarded. Usage of TTL parameters may cause false-positive errors, i.e. Even if the searched resources exist and are available on the Grid, the system may return unsuccessful results to the queries because the TTL limit is reached. This is especially a problem when searching for rare resources that are only present on a limited number of peers. Moreover, scalability of unstructured solutions may be limited by the exponential growth of flooding-related traffic. To address this problem, more efficient flooding methods have been subsequently developed, e.g. selective flooding [26], random walks [27], routing indices [28], semantic overlays [29] , [30] and etc.

4.2.2.2.3. Structured P2P architecture:

For eliminating the issues (e.g. flooding and false positive error) suffered by the unstructured p2p, the structured solutions propose the construction of topologies with strict properties that allow deterministic resource discovery. On these systems, referred to as Distributed Hash tables (DHT) [31], the overlay network is organized in such a way so that information can be easily located by means of keys associated to the underlying network node identifiers. The DHT based P2P systems such as Chord[31], Pastry[32], Tapestry[33] and CAN[34] are efficient and scalable, while there are limitations on the characterization of indexed resources. Specifically, Grid RD queries do not typically match to a unique key, as required by DHT, therefore introducing a degree of difficulty in locating complex queries (i.e. multi-attribute range queries). Overcoming the above shortcomings, several peer-to-peer (P2P) schemes, e.g. extended the DHT scheme to support multi-attribute rang query while achieving satisfactory results in terms of resource discovery efficiency, for example MAAN[35], SWORD[36] and NodeWiz[37] have been proposed to index and discover Grid resources in a structured P2P network. Additionally Mercury[38], and other have been proposed to support multi-attribute range query but without using DHT structure.

5. Survey of Grid RD systems: discussion:

Before engaging in a detailed analysis of resource discovery approaches, it is imperative that the range of metrics in which we will use it to analyses the studies of resource discovery in grid is fully understood. In the following subsection we discussed the metrics which using to evaluate the performance of approaches for grid RD.

- ***Performance Metrics of Resource Discovery Problem***

a) Complexity

Is a basic measure, which determines the run-time of the algorithm. It is considered in two aspects, message and time complexities. The *message complexity* deals with the number of transferred messages. Relatively higher message complexities may result in congestion in the network, which may negatively affect the performance of the algorithms. On the other hand, *time complexity* determines how many steps are required for the termination of the algorithm.

b) Scalability

Scalability in Grid is the increasing number of resources or users that use these resources and this is a very important measure, because Grids are large-scale environments in their nature. The performance of a system, which is not scalable, degrades very rapidly as the size of the environment grows. This fact may cause the algorithm to perform poorly in such environments.

c) Dynamicity

Is another important factor in analyzing Grid algorithms since nodes in Grid systems might be highly dynamic in terms of joining and leaving the system, mostly without any notice. The algorithms that tolerate the dynamicity of the environment are more suitable for Grid systems.

d) Reliability

Is also an important measure because in some cases, erroneous query results may cause irrecoverable faults. For instance, RD algorithms, which may result in false-positive errors might not be suitable in Grid systems. Also using architecture that is considered single point of failure will be has highly affected in reliability of grid system.

In addition; Support for ***multi-attribute, dynamic attribute*** and ***range queries*** is a decisive criterion on selecting the methodology in most cases since the running applications may require those types of queries.

5.1. Agent-based Grid RD

Rahimzadeh el al. [39] proposed RD system for economic grid. Their proposal modeled Grid as a graph, every node of which represents a resource agent and each one of these agent carries a specific resource with a presenting expense called "Price". Resource agents interact with their neighboring nodes to form a chain of resources required for executing the tasks. In order to start the proposed algorithm, a node should undertake the responsibility of searching the resources. This node should be selected randomly and meet one of the required resources with a cost less than the requested budget, and this node will call Admin node, after that the Admin node use a messaging system approach for tries to meet all the required resources given the requested price. Authors use semantic similarity function know association degree between resource types required for task and resource agent. Since this algorithm need to choose Admin node randomly, the flooding is required and thus, message complexity in the worst-case scenario, the query passes through all the edges between nodes, the worst-case message complexity of this algorithm is *O(E)* and the time complexity is *O(D)* where E is the number of edges and D is the diameter of the agent graph. Here we notice that messaging approach suffer from high number of messages to discover all the requires resources. Admin node may consider single point of failure if it fail before responds to the user with all requested resource. The algorithm does not contain any bottlenecks. Moreover, since flooded requests do not have a *TTL* limit, false-positive errors do not exist. The algorithm supports all multi-attribute and range queries the processing of query happen by agent resource without any hashing.

Wang et al. [40] propose RD in economic grid. In their scheme, the provider and requester send its information to their corresponding discover agent (DA). Each DA

posses Resource Status Table (RST) which include status and specification of resource which represent. When node need to search about specific resource, it send information about resource to corresponding DA Which in turn will look for them in RST, if there has match resource, it send message result to its user, else it send the request to another discovery agent in hierarchical architecture. Authors also propose negotiation framework which use to make RD more efficient if discovery failed. The idea involved in the negotiation as follow: when discovery reach the root of hierarchy without result, then agent will carry the negotiation among the resource provider agent (Which has the closest match to the required resource) and grid user (requester) by the Grid architecture for Computational Economy (GRACE) with Resources Pricing Fluctuation Manager (RPFM) middleware. They claim that this feedback model plays a very important role in the agent-based system when resource discovery failed for cost bound, as well as, negotiate solution will let Grid users use resources effectively and the resource providers can get the maximize investment and profit. Time and message complexities for this algorithm is *O(A)* where *A* is number of resource agents. Since agents update the detail of its representative for the resources in the RST, algorithm supports dynamicity. The algorithm does not contain any single point of failures, but the discover agents at the high level of hierarchical may cause bottleneck if high number of low level queried them at the same time. On the other hand, since whole resource information exists in RST without any hashing, the system supports range and multi-attribute queries.

Sotiriadis et al. [41] propose a theoretical approach of decentralized agent models based on Self Led Critical Friends (SCF) method, since this study concentrate on communication between loosely connected virtual organization (inter-operable Grid). Authors in this study concluded that the most appropriate way to resource discovery in large grid is divide the grid to multiple VO. The *Internal Broadcasting Agents* method utilize internal connections and transmit data to all individuals within a VO. Each part will perform communication and delegate job to any connected node, while several agents are able to travel through the domain and exchange information. They indicate that there are two common problem in this approach as a SCF agent may be fails and may be loss communication with an inter-connected VO agents, to solve this problems, authors suggest providing with a communication between each subset of decomposed domain agents and member of the domain.

Singh et al. [42] propose RD based on Multi Agents based on the unstructured P2P network model and use similarity of content shared by peer and exact keyword match for RD, peers clustering based on the conceptual content of resources shared by peer. *Singh et al.* suggest in their study four agents collaborating on each peer are: Interface Agent IntA; Local Agent LA; Information Agent InfA which holds information about peers that are semantically similar to this peer, and Reconnaissance Agent which is a mobile agent that is created by the IntA upon user's search request. RA migrates to new

peers by requesting node address from InfA. RA's task is to migrate to peers and to investigate LA that is responsible for hosting resources (hence keywords) about their possible similarity to user's query and report it to IntA. The proposed system consists of a bootstrap server and agent management service (AMS). The bootstrap server maintains a list of peers that are currently in the system, upon registration/joining, the bootstrap server replies with list of peers that are semantically similar to this peer. On the other hand, AMS is server which informs RA to find the container/platform where the selected peer is located. In this algorithm, the complexity is low because each InfA has information about all similarities which shared by other peer, thus complexities will be depend on number of peers needed which host the similar resource which is required. Update InfA with all information about peer may be effect negatively of scalability. The queries are processed within resource nodes in this system, thus, this eliminates bottleneck problems in the relaying nodes, but since the bootstrap server and AMS is centralized, it might become a bottleneck and a single point of failure . And since the query agents are migrated according to a planned strategy, false-positive errors do not exist. The queries are processed within nodes without any hashing; so, the algorithm supports dynamic- attribute, range and multi-attribute queries.

Tan et al. [19] propose scheme to disseminate Grid resources and spatially map them on the Grid according to their semantic classification. The scheme exploits the random movements and operations of a number of Mobile Agents (MA) that travel the Grid using the P2P interconnections. In this scheme each peer is connected to neighbor peers, including horizontal, vertical and diagonal neighbors. Dissemination of resources happen as follow: in each class there has agent move randomly and when gets to a Grid host, if it is currently unloaded, it must decide whether or not to pick the resources of class that are managed by the current host. The probability of picking the resources of class is defined through by a pick random function. Whenever an agent specialized in a class gets to a new Grid host, it must decide whether or not to drop the resources of class, in the case that it is carrying any of them. Discovering of resource happen as follow : a query message first travels the Grid network with a blind/random mechanism; however, the search procedure is turned into an informed one as soon as the query approaches a low entropy region, i.e. a region which has gathered resources belonging to one particular class. In this algorithm, time and message complexity are *O(C)+O(1)*, where C is number of semantic classes (clusters). Each peer loads by information of all class peers, in a highly dynamic and large-scale Grid system, the high number of update messages could limit the scalability of the algorithm. This system does not suffer from bottleneck or single point of failure. On the other hand, The algorithm supports all multi-attribute, dynamic-attribute and range queries since the queries are resolved within the nodes without any hashing function.

Fattahi and Charkari [43] propose a MA technique based on unstructured peer to peer model and this scale-free network topology produces using *Barabasi and*

Albert model [44]. MA modifies the rout table during the migration based on Ant Colony System (ACS). Thus central control eliminated and node autonomy provided in this algorithm. Each node in their overlay store its resource information, in fact node in this grid topology represent a MDS. In result, the message and time complexities of this algorithm are *O(N)* where *N* is the number of resource nodes in Grid. One of the main restrictions of this approach is the simplicity of query which is defined with only one attribute and this constraint make this algorithm does not support multi-attribute query. On the other hand, since flooded requests do not have a TTL limit, false-positive errors do not exist. The algorithm supports all dynamic-attribute and range queries since the queries are resolved within the nodes without any hashing function.

5.2. Query-based Grid RD

5.2.1. Centralized-based Grid RD

First implementations of a Grid Information System GIS used techniques based on centralize directories, which are used by Globus MDS(LDAP). Advantage of this approach is supporting global system information, and hence the client requester makes the selection decision based on global state information; as well as grid resource discovery using centralized systems provide grid middleware developers an easy to use interface to manage grid resources. They keep grid resource discovery information by using centralized databases. The centralize systems support a multi-attribute rang query 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 centralized RD systems do not support dynamic-attribute queries efficiently. In a large scale grid environment, the centralization of the service may has several design issues including: (i) highly prone to a single point of failure; (ii) lacks scalability; iii) high network communication cost at links leading to the information server (i.e. network bottleneck, congestion); (iv) the machine running the information services might lack the required computational power required to serve a large number of resource queries and updates; and (v) problems of keeping indexed information up-to-date.

Given the disadvantages described above, the centralized approach is inappropriate to RD in large scale grid because it suffer from scalability as grid has become largest and lack for reliability. thus there have not studies at the most recent years use this approach and all recent studies use decentralized approach as it is clear from remainder of this section. A current trend is use decentralized techniques which overcome most of centralize issues.

5.2.2. Decentralized-based Grid RD

5.2.2.1. Hierarchical architecture

Kovvur et al.[45] proposed a model for RD which extension the ideas of the adaptive push and pull method. The grid environment in this model consists of three layers; *Main coordinator* (layer-1), *Aggregators* (layer-2) and *Grid nodes* (layer-3). Each Grid node in the environment runs a daemon that collects state information such as CPU Speed, CPU loads, Memory size and the available memory. Each *Aggregator* maintains a database of aggregate state information of these sub-resources (i.e. VO). Nodes send their resource information to the resource *Aggregator* periodically, instead of directly sending this information to a *Main coordinator*. The daemon at main coordinator pulls the aggregate state information from the *Aggregator* and respond to queries from the scheduler. Complexity of this algorithm is *O(A)* since *A* is the number of aggregators. The main drawback of this scheme is that *Aggregators* and *Main coordinator* is centralized, they might become a bottleneck and a single point of failure, thus, scalability will be negatively affect when grid become larger. Moreover it doesn't support dynamic attribute efficiently since the resources are advertised to aggregators by periodic updates. On the other hand, it support multi-attribute range query because queries process in *aggregators* or *Main coordinator* without any hashing.

Kocak and Lacks [46] propose placing the load of managing the network RD on to the network itself inside of the routers by programmable networking hardware with using tables in routers for resources information similar to routing table. Their system consists of *Provider (*or *Consumer)* nodes, *Routers* and *VO Host*. *Kocak and Lacks* propose five phases involved in the lifetime of a Grid RD: *Subscription*, *Advertisement*, *Transaction*, *Sign-off*, and *Retirement*. Before the *Subscription phase*, the resource provider acquired software from the VO. During the *Subscription phase*, the resource provider is subscribed to the *VO's Host*. During this transaction, an account is setup that includes ways for the *VO's Host* to track the trustworthiness of the resource provider; the second phase, *Advertisement*, is when the grid resources advertise their availability to its *Router*. The *Transaction*, the third phase, is when the actual task is delivered to via a discovery process, computed on, and published from the resource providers. For the *sign-off* phase, the grid is notified that the resource is unavailable for an unknown period of time. Discovery happens when user send request about specific resource to *VO Host* Which in turn to one of routers in VO and then request travels from router to router until find resource which fulfill requirement. In this scheme, one of the drawbacks is that *Routers* and *VO Host* might be single point of failure and bottleneck, as result, scalability may be negatively affect. Complexity of this proposal is *O(R)+O(1)* which *R* is number of Routers in Grid. Moreover, this scheme doesn't support dynamic attribute efficiently since the resources are advertised to Routers by periodic updates but support

multi-attribute range query since queries are processed within Routers without any hashing.

Chang & Hu [47] propose a hierarchical (Tree) overlay for resource discovery. They use more than one indexing server (IS) to store the information about resources. Bitmaps is used to represent resource attributes and the resource information is aggregated from leaf nodes to thier ancestors. Each *IS* in the RD tree has two bitmaps (local and index bitmaps) and one counter array to store the resource information. The local resource bitmap stores the local resource information. In a leaf node, only the local resource bitmap is necessary. The index bitmap stores the resource information of a node's children since it produce by the bitwise OR operation of its children nodes' bitmaps. The number of positions in the counter array is the same as the number of bits in the bitmap. Each position in the counter array records the total number of resources in the children nodes. Here not all queries have to go through the root, also the root only has to handle information from its next level childern only, not all of its childerns. A query can also be expressed as a bitmap, called as a query bitmap. A bitwise AND operation is used to check for matches. Space complexity of this scheme is *O(BARN +DN), and* time Complexity is *O(B log N);* where *N* is number of nodes, *R* is number resources in the system, *A* is number of attributes which resource is consisted and *B* is the length of the bitmap by bits. Although this approach decrease the number of nodes which will be passed but still there are many unnecessary path will be passed. In this system indexing *server* might become a bottleneck and a single point of failure, thus, scalability may negatively affect. On the other hand, this algorithm doesn't support dynamic attribute efficiently since the resources are advertised to super-peers by periodic updates and it don't support multi-attribute range query efficiently because simplicity of bitmap query.

Khanli & Kargar [48] present scheme for RD extension to previous work [3] use a weighted tree for resource discovery (footprint RD tree). They use footprint to access the directly appropriate resource without visiting additional and unnecessary nodes and no time is consumed. This algorithm solves the traffic problem but it is still result some time high traffic without finding the required resource special in case of multi-attributes query. Space complexity of this scheme is less than *O(BARN +DN)* where *N* is number of nodes, *R* is number resources in the system, *A* is number of attributes which resource is consisted and *B* is the length of the bitmap by bits. Time Complexity is less than *O(B logN).*

5.2.2.2. Super-Peer architecture

Gallardo et al. [49] propose unstructured topology based on Hypercube *HGRID.* An *n* dimensional hypercube (H_n) has $V(H_n)= 2^n$ nodes, where each one represents a Grid Information System *(GIS)* and each vertex (*GIS*) has exactly *n* neighbors, since each VO

provides its resources to one of *GISs*. Each node (or *GIS*) has an identifier that goes from 0 to 2^{n-1}. Two nodes are said to be directly connected to each other (they are said to be neighbors in the *m-th* dimension) if the binary representations of their identifiers differ exactly by the *m-th* bit. The service discovery is tried first inside the requester's own *GIS*, If there is no provider, then the request is redirected to other *GISs*. *Gallardo et al.* improve searching algorithm which had been done by [50] to make it address non-alive node. The queries are processed within central *GIS* at VO. Thus *GIS* might become a bottleneck and a single point of failure. The worst-case message complexity of the algorithm is O*(S)* where *S* is the number of *GISs*. The time complexity is *O(N)* where *N* dimensional of hypercube. Moreover, this algorithm may suffer from false positive errors since it work without TTL. The queries are processed within nodes without any hashing; so, range and multi-attribute queries are supported, but since the update of dynamic resource attributes are held in discrete intervals, this system do not support dynamic-attribute queries efficiently. One of the drawbacks of existing routing is multiple enquiries for non-alive neighbor from more than one of its live neighbor, as a result, high number of messages may be negatively affected in scalability.

Miriam and Easwarakumar [51] present RD based on Hypercubic P2P Topology *HPGRID* which is the hypercube structure with additional neighborhood links. This work is modifying for algorithm which presented by *Gallardo et al.* [49]. The *HPGRID* nodes are partitioned isomorphically listing the available resources according to their zones which aids the user to select the needed resource to execute its job rather than traversing the whole grid nodes. In *HPGRID*, each node represents a Cubic GridPeer *(CGP)* where each CGP is a collection of Grid Nodes *GNs*. Here, *CGP* is equivalent to a super node. Each *CGP* controls the access of a group of local computing resources and the P2P mode are adapted to interact the information between *CGPs*. When the users need to searching about resources, they first query the resources in the domain of *CGP*. If no query result, the search will be carried out through *CGP* to query the other *CGP* with P2P way. *HPGRID* system uses *Parameterized HPGRID* algorithm which reaches all the alive nodes with minimum number of hops. The queries are processed within CGP, Thus *CGP* might become a bottleneck and a single point of failure. The message complexity of this algorithm is $O(\log^{N}_{D})$ where *N* is number of nodes in Hypercubic and time complexity is *O(D)* since *D* is dimension of hypercube overlay. The queries are processed within nodes without any hashing; so, the algorithm support multi-attribute range queries, but since the update of dynamic resource attributes are held in discrete intervals, this system do not support dynamic-attribute queries efficiently.

Khoobkar and Mahdavi [52] propose developing of another generation of systems for discovery of resources based on Routing Indexes (*RI*) technique. *Khoobkar and Mahdavi* address the limitation of previous work presented by *Mordacchini and Orlando* [53] which is use fixed *RI* to solve RD problem. They propose multi-level

overlays that are combines architecture of hierarchical and super-peer approaches. In lowest level, i.e. presents city networks (or smaller networks such as LAN), attribute space is distributed (cleaved) between *N* super peers and each super peer is responsible for maintaining information that their range fall into the range of the super peer's responsibility. Peers send their information dynamically to the responsible super peer. In the upper level (country network), one or multiple super peers of each definite range are selected as delegations and connect with delegations of neighboring countries (same range). These connections continue in upper levels. These connections are enabled by a variable *RI*. Therefore, any received query is processed in the relevant super peer locally, according to its range. Then, the next level's *RI* (city level *RIs*) is checked and this hierarchical routing will continue in the upper levels, if nothing is found on the lower levels. In this system super-peers might become a bottleneck and a single point of failure, thus, scalability may negatively affect. The message and time complexity are less than *O(S)* and *O(D)* respectively, since *S* is number of super-peers and *D* is diameter of overlay in highest level. On the other hand, this algorithm doesn't support dynamic attribute efficiently since the resources are advertised to super-peers by periodic updates but support multi-attribute range query.

Vashisht and Sharma [54] propose another generation of systems for discovery of resources, in order to improve the search complexity of the resources from the grid information system. In this scheme, nodes classify as *manager (Ultra-Peer), executer (Service Provider)* and *user* who request for a resource. The selection of *Ultra-Peer* is done on certain criteria like node should be static, node should have maximum bandwidth and etc. For each resources in grid there is one *Ultra-Peer* node, where the registry of the resources is done, and resources is treated as local to the registry node. After that selection of *Ultra-Peers* are made according to the capability of the node. The leaf nodes with similar services are grouped together so as to ease out the searching. Communications between the *Ultra-Nodes* are peer to peer, which follow ring topology. These resources are characterized according to their properties and they are stored in the form of tree using LC-trie (Level Compressed Trie) 55]. LC-trie is an improved of PAT (Patricia or Path Compressed Trie) [56]. They improve PAT by changing branching factor of some nodes if some certain conditions are satisfied. If the *Ultra-Peer* nodes fail at some point of time, then the nodes will broadcast the message in the network for finding another *Ultra-Peer*. This algorithm doesn't support dynamic attribute efficiently since the resources are advertised to *Ultra-Peer* by periodic updates but support multi-attribute range query since queries are processed within nodes without any hashing. The message and time complexity are less than *O(U)* since *U* is number of *Ultra-Peers* and since this scheme uses concept of caching. On the other hand Ultra-Peers might become a bottleneck and a single point of failure, as a result, scalability may negatively affect.

5.2.2.3. Unstructured P2P architecture

Hu and Zhao [57] present a Center-Distributed Virtual community model for RD. *Virtual Community (VC)* is a user set who adopt a same kind of Grid recourses. As a member of a *VC*, he held or will hold the information about common shared resources in the *VC*. Each community can be mapped to a circle (or ring). When a new node wants to join the *VC*, it must firstly connect a node of the community, then the connected node will distribute a unique member *id* to this new node. When a node wants to release the query, it respectively chooses the clockwise and counterclockwise direction to send message to his two adjacent nodes. The adjacent nodes should continue to passed the message, in order to make sure the message diffused in a clockwise or counterclockwise direction. We notice that this paper deal with one virtual community only but doesn't solve how query will transmit between multiple *VCs*. Thus this approach suitable in small world since it aims to decrease the time complexity in the same *VC*. In this algorithm, node of the community might become single point of failure. To apply this algorithm for whole *VCs* in Grid, then time and message complexity will be $O(N^2)$ since N is number of nodes. The queries are processed within nodes without any hashing; so, range and multi-attribute and dynamic queries. In this algorithm, the distribution of queries is not limited by a TTL value, which eliminates false-positive errors.

5.2.2.4. Structured P2P architecture

Sun el al. [58] proposed overlay for nodes in Grid includes three layers based on three different peers: The *Super peer-agent*, *Super peer* and *Ordinary Peer*, each VO corresponds to a super node (*Super peer*), and each peer must register in *Super Peer*. The *Super peer-Agent* is resource service agent of the region, and can take charge of one or more similar Super peers and it play the active role of coordination in the information search and load balancing. each *Super peer* saves information about adjacent peer, plays dual role as client and server, which not only accepts and updates information from peer or other *Super peers*, but also transmit its information to *Super peer-Agents*, peers and other *Super peers* across VO, each peer interconnects as neighbor mutually, exchanges, transfers and checks message by way of P2P. This scheme employs DHT (Distributed Hash Table) to indexing and searching, and it use TF_IDF (Term frequency_ inverse document frequency) technology to generates keyword. The process of resource discovery is that user peer firstly forward query in its respective VO, if failed or information needed did not match, the Super peer will turn to other *Super peers* or *Super peer-Agent* for help. Since DHT support only single key. Thus this algorithm doesn't support rang and dynamic attribute queries efficiently. Moreover, since *Super peer-agent* is centralized, it might become a bottleneck and a

single point of failure. There is two updates (one that will be transmit to *super peers* and another update that will be send to neighbor peer), thus, these updates will result high traffic in network and this will be affected at scalability. The complexity of this algorithm is *O(log N + log S)* where *N* is number of peers in VO and *S* is number of Super-Peers in Grid.

Pipan [59] propose different overlay named *TRIPOD* that builds upon the existing research on triangulated and tree overlays. This hybrid overlay tree provides a global connectivity. In the case of a failure, the triangulation overlay allows the tree to be rebuilt by executing efficient proximity queries to substitute for the failed node. This proposal uses divide & conquer algorithm. RD protocol consists of two processes; where the first is a preprocessing that is located on a resource which calculates the Bloom filters, and the second is the routing mechanism. When node requests resource, then the first step checks whether the resources of this node already satisfy the query requirements and if so push the node into the stack of suitable resources. If the requested amount of suitable resources is met, the answer is sent to the sender of the query. Otherwise, query is forwarded to all the children nodes (excluding the sender node). In the case that none of the children met the query requirements, forwards the query to the parent node. If the sender of the query was the parent node, then the reply event is set in the query. This algorithm produce high traffic message when update bloom filter since when attributes value of one node in neighbor tree change, this changing must send to all neighbors' parent tree, as a result, scalability will be negatively affected. As the resource information, which is used for matching, is stored at the resources themselves, thus this algorithm enable the use of the dynamic resource information and support multi-attribute range queries**.**

Lin et al. [60] propose a self-adaptive resource index and discovery system (SARIDS) to achieve load balancing. *SARIDS* adopts a two-tier (i.e. intra and inter Overlay) architecture based on the structured P2P overlay. The intra-overlay is constructed by normal peers with the same attribute via the locality preserving hash function and Chord [4] is applied as routing infrastructure. On the other hand, the inter-overlay is constructed by super-peers with classified attributes in different intra-overlays. When a query or publishing information is issued, it starts to route to specified attribute. Then, the request is routed to the dedicated intra-overlay through that super-peer and locates the normal peers with the matched values. They use live-rejoin-style protocol to support intra-overlay load balancing and use the sequential searching solution to resolve the range queries. Here, to support the multi-attribute range queries, the randomizing hash function and the locality preserving hash function are adopted to map the attribute information to specific normal peers and to overcome the non-uniform data distribution among peers and the load imbalance, they also propose load balancing mechanisms to balance the load of normal peers in the intra-overlay and redistribute normal peers among different intra-overlays. *SARIDS* uses the bidirectional searching

method to resolve range queries. In *SARIDS* the number of routing hops for finding the super-peer with the specified attribute in the inter-overlay is $logN_1$, where N_1 is the size of an inter-overlay. On the other hand, the number of routing hops for locating the normal peer with the matched value is $logN_2$, where N_2 is the size of the specified intra-overlay. Thus, complexity of this algorithm is $logN_1 + logN_2 + k$ where k is the number of routing hops of the bidirectional searching method. On the other hand, the queries are routed using Chord mechanism; therefore, false-positive errors do not exist but super-peers might become a bottleneck and a single point of failure**.** Moreover, the algorithm supports both range queries, multi-attribute queries but don't support dynamic-attribute queries efficiently**.**

6. Evaluation and analysis:

6.1.Semantic-based Grid RD

Given that semantic technology can be setup as layer over centralized or decentralized overlay. Thus with respect to this technology we will discuss the evaluation of its performance only without present it as subsection in section (5) because there are several research have been presented in this survey such as the studies have been discussed in subsection (5.1) i.e. Agent based approach. The main issue in semantic RD systems is that each work focuses in description of resource attributes. This means a system may meet only one or two of the grid RD requirement. Requirements include interoperability, scalability, decentralization and dynamism. Interoperability here means the ability to span multiple administrative domains in discovering the resources and services. With respect interoperability there is issue which is semantic interoperability or in other words, how to ensure that two concepts from two different ontologies are referring to the same resource that they represent. Therefore, using different ontologies do not bring the system closer to achieving interoperability, and thus it requires using common ontologies in service descriptions in order to reach semantic agreement. The remain three requirements are much related to indexing or registration and discovering of semantic information of resources and this mechanism depend on approach which will be use (i.e. centralized or decentralized approach). Another issue related to semantic technology is high computation that is required to compute concept similarity and compare user request with all those concepts.

6.2. Symmetric-based Grid RD

We described and analyzed several grid RD studies, which are presented by using different approaches. Although each study has its own advantages and disadvantages, commonalities can be classes clearly distinguished when they are examined in their own classes. Regarding those commonalities, an evaluation can be easily performed between

five different architectures of symmetric-based approach. A summary of evaluation can be seen in Table 4 .

***Table (4):** Summary evaluation of symmetric-based Grid RD approaches and methods*

Query	Definition
Agent-based approach	In agent-based grid RD approaches in which the underlying network topology is unstructured, the system does not suffer from bottleneck problem. The main factor that affects the scalability of the system is diffusion technique of the requests. When agents are used, the diffusion is handled by using random approach, which is unscalable because of its message complexity. On the other hand, if MA are used, because of their autonomy and self-decision properties, more clever routing techniques are applied to increase the scalability [7]. On the other hand, when the Grid is dynamic, since in agent-based approaches flooding is used to distribute the queries, dynamicity of the nodes does not perturb the dissemination of queries. But when the MAs are used, the queries are routed on a single path, and the failure on any node on this path may cause loss of queries in the network. The schemes which are presented in this approach do not have single point of failures in general since central managers do not exist. Moreover, the distribution of queries is not limited by a TTL value, which eliminates false-positive errors. But, we believe that in large-scale environments in which an unstructured network topology exists, TTL values are essential to avoid extremely long query response durations.
Centralized-based approach	The centralized systems based Grid RD provide global system information, and hence the client requester makes the selection decision based on whole system global state information, but In a large scale grid environment, the centralization of the service may easily create bottlenecks on the central servers. The bottleneck problem may arise. 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. Time and message complexity is *O(1)*. Centralized approach support the multi-attribute and range queries since the resource information is stored in central databases without any hashing of resources information. Because the resource information is send to central directory at periodic intervals, in this method, dynamic-attribute queries in its nature does not support.

Decentralized-based approach:	Distributed based model suffers from lack of global system information, and hence the client requester makes the selection decision based on partial system global state information. Furthermore, since Distributed based models do not maintain centralized resource management method, the system state may become imbalance if many grid clients greedily select the same resource providers that have the highest computational power. On the other hand, the flat indexing structures pose a major challenge to the global resource monitoring in Grids due to its large-scale and decentralized nature.
Hierarchical architecture	The hierarchical systems based Grid RD algorithms provide a more scalable platform than the centralized ones and still provide a simple user interface to manage grid resources[6]. 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 studies in this approach support the multi-attribute and range queries since the resource information is stored in databases which are capable of processing complex queries. Since, in many studies, the information is not verified in the resource nodes after querying databases, thus hierarchical approach don't support dynamic-attribute queries efficiently.
Super-Peer architecture	Most super-peer-based P2P algorithms use flooding between the super-peers. Decreasing the size of the flooding domain reduces time and message complexities use algorithms of flooding. Even these types of mechanism can be considered as more scalable than unstructured systems; super-peers may suffer from being bottlenecks in the system when the number of requests is large. Moreover, the super-peers are responsible for a set of resources and failure of a super-peer will break the imaginary connection of the resources, which exist are available. Therefore, dynamicity of a super-peer badly affects the domain of the queries. This fact also negatively affects the reliability of this approach by turning super-peers into single point of failures. However, since the queries are resolved by super-peers by checking the index tables, this approach supports range queries and multi-attribute queries easily. But, because the resource information is collected by the super-peers at periodic intervals, this method dynamic-attribute queries in its nature does not support.

Unstructured P2P architecture	Considering the nature of the unstructured P2P RD techniques, in most cases, because of the common routing mechanisms, the complexity of the algorithms is high, which makes the approach unscalable. The message and in some cases time complexities have higher order of growth than the scale of the network. Nearly all unstructured systems suffer from false-positive errors caused by the usage of TTL limitations. Even if the searched resources exist and are available on the Grid, the system may return unsuccessful results to the queries because the TTL limit is reached. Otherwise, when TTL is set to a higher value, asymptotic increase in the messages negatively affects the bandwidth and runtime of the algorithms. On the other hand, this approach can easily handle dynamicity of the Grid since both resources and indexing nodes are distributed to the entire network. In any case, even if the network is very dynamic, queries are not lost in the network and propagation of the queries continues until a TTL value is reached.
Structured P2P architecture	Structured P2P systems are more scalable than unstructured ones, in terms of traffic load, but need to have strong self organization capabilities in order to be able to maintain their rigid structure. Structured systems are prone to node failure, and unpredictable node departures. Although in the past few years considerable effort has been devoted to research on structured P2P systems, they have also earned a lot of criticism for their high maintenance cost in the presence of high churn [5]. Since these studies use topological structures, time and message complexities of the algorithms are around *O(log N).* even if the nature of structured P2P-based RD algorithms do not support range, multi-attribute and dynamic-attribute queries, nearly all studies, which are developed in this scope, find reasonable solutions to support all different types of queries.

7. Conclusion:

Resource Discovery is the process of finding the satisfactory resources according to the user's request, including resource description, resource organization, resource lookup and resource selection. Supporting multiple types of resources, high performance, and massive scalability are some of the most important goals in the design of a distributed resource discovery system. We must pointed some of notices about approaches and methods of Grid RD as follow:

- Agent-based RD approach are suitable for dynamic middle-scale Grid environments in

which some false-positive errors are acceptable, it brings the advantage of having up-to-date resource information all the time.
- 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.
- Hierarchical methods are suitable for middle-scale Grid networks 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 networks .
- Super-Peer-based grid RD architecture are suitable for middle-scale Grid networks in which reliability of super-peers is strictly provided. And they are not suitable for dynamic-attribute queries and if the false-positive errors cause serious problems.
- Unstructured P2P-based grid RD systems are suitable for small scale, highly dynamic Grid environments.
- Structured P2P-based methods are suitable for large-scale Grid systems in which reliability is important and dynamicity is low.

References:

[1] I. Foster, *The Grid: Blueprint for a New Computing Infrastructure*, 2002.
[2] K. Krauter, R. Buyya, and M. Maheswaran, *A taxonomy and survey of grid resource management systems for distributed computing*, Software Practice and Experience, Vol. 164, No. 2, P. 135-164, 2002.
[3] A. Sharma and S. Bawa, *Resource Discovery using Ontology Approach in Grid Environment*, Challenges & Opportunities in Information Technolog y(COIT), P. 63-67, 2007.
[4] K. Karaoglanoglou, *Discovering Resources and Mapping in Large-Scale Distributed Environments*, October (2009).
[5] P. Trunfio et al., *Peer-to-Peer resource discovery in Grids: Models and systems*," Future Generation Computer Systems, Vol. 23, No. 7, P. 864-878, Aug. 2007.
[6] A. Hameurlain, D. Cokuslu, and K. Erciyes, *Grid Resource Discovery Based on Centralized and Hierarchical Architectures*, International Journal of Metadata, Semantics and Ontologies, Vol. 3, No. 1, 2010.
[7] A. Hameurlain, D. Cokuslu, and K. Erciyes, *Resource discovery in grid systems□: a survey*, International Journal of Metadata, Semantics and Ontologies, Vol. 5, No. 3, P. 251 - 263, 2010.
[8] M. Aktaruzzaman, *Literature Review and Survey□: Resource Discovery in Computational Grids*, P. 1-71, 2003.
[9] M. Sedaghat, M. Othman, and M. N. Sulaiman, *Agent's role in grid environments during resource discovery: A review*, In ICRCCS h'09 Proceedings of the International Conference on Research Challenges in Computer Science, IEEE Computer Society Washington, DC, USA, P. 1-9, 2008.

[10] M. I. Hassan and A. Abdullah, *Semantic-Based Grid Resource Discovery System A literature review and taxonomy*, In Information Technology (ITSim), International Symposium , Vol. 33, P. 1286-1296, 2010.

[11] A. K. Shaikh, S. M. Alhashmi, and R. Parthiban, *A Semantic Decentralized Chord-Based Resource Discovery Model for Grid Computing*, IEEE 17th International Conference on Parallel and Distributed Systems, P. 142-148, 2011.

[12] F. Heine, M. Hovestadt, and O. Kao, *Towards Ontology-Driven P2P Grid Resource Discovery*, In Proceeding GRID '04 Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing, P. 76 - 83, 2004.

[13] B. Chandrasekaran, J. R. Josephson, and V. Benjamins, *What are ontologies, and why do we need them?*, IEEE Intelligent Systems, Vol. 14, No. 1, P. 20-26, 1999.

[14] *http://www.semanticweb.org/* .

[15] A. Maedche and S. Staab, *Measuring Similarity between Ontologies*, Knowledge Engineering and Knowledge Management Ontologies and the Semantic Web, Vol. 2473, P. 251-263, 2002.

[16] E. Bagheri, M. Naghibzadeh, and M. N. Bagheri, *A New Approach to Resource Discovery and Dissemination for Pervasive Computing Environments Based on Mobile Agents*, Scientia Iranica, Vol. 14, No. 6, P. 612-624, 2007.

[17] A. Yousif, A. H. Abdullah, M. Shafie, A. Latiff, and M. B. Bashir, A *Taxonomy of Grid Resource Selection Mechanisms*, International Journal of Grid and Distributed Computing, Vol. 4, No. 3, P. 107-118, 2011.

[18] V. Iyengar, S. Tilak, M. J. Lewis, and N. B. Abu-Ghazaleh, *Non-Uniform Information Dissemination for Dynamic Grid Resource Discovery*, Third IEEE International Symposium on Network Computing and Applications *(NCA).* Proceedings, P. 97 - 106, 2004.

[19] Y. Tan, J. Han, and Y. Wu, *A Multi-agent Based Efficient Resource Discovery Mechanism for Grid Systems*, Journal Of Computational Information Systems, Vol. 11, P. 3623-3631, 2010.

[20] I. Foster and N. R. Jennings, *Brain Meets Brawn: Why Grid and Agents Need Each Other Agent-Based Computing*, Information Sciences, 2004.

[21] S. Fitzgerald et al., *A Directory Service for Configuring High-Performance Distributed Computations*, In 6th IEEE Symposium on High-Performance Distributed Computing, P. 365-375, 1997.

[22] *Condor Project: http://www.cs.wisc.edu/condor/*.

[23] X.Schopf and J.Zhang, *Performance analysis of the globus toolkit monitoring and discovery service(mds2)*, In Proceedings of the International Workshop on Middleware Performance (MP 2004), 2004.

[24] I. Foster and A. Iamnitchi, *On Death , Taxes , and the Convergence of Peer-to-Peer and Grid Computing*, In 2nd International Workshop on Peer-to-Peer Systems (IPTPS'03}, Vol. 2735, P. 118-128, 2003.

[25] A. Iamnitchi and I. Foster, *On Fully Decentralized Resource Discovery in Grid Environments*, In Proceedings of the Second International Workshop on Grid Computing, P. 51-62, 2001.

[26] S. Arunkumar and R. S. Panwar, *EFFICIENT BROADCAST USING SELECTIVE FLOODING*, In INFOCOM '92. Eleventh Annual Joint Conference of the IEEE Computer and Communications Societies, IEEE, P. 2060-2067, 1992.

[27] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker, *Search and replication in unstructured peer-to-peer networks*, ACM SIGMETRICS Performance Evaluation Review, Vol. 30, No. 1, P. 258, Jun. 2002.

[28] A. Crespo and H. Garcia-molina, *Routing Indices For Peer-to-Peer Systems*, In 22nd IEEE Int. Conf. on Distributed Computing Systems, P. 84-95, 2002.

[29] A. Crespo and H. Garcia-molina, *Semantic Overlay Networks for P2P Systems*, Technical report, Stanford University, 2002.

[30] C. Tang, Z. Xu, and S. Dwarkadas, *Peer-to-Peer Information Retrieval Using Self-Organizing Semantic Overlay Networks*, In SIGCOMM '03: Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, ACM, New York, NY, USA, P. 175–186, 2003.

[31] I. Stoica et al., *Chord: a scalable peer-to-peer lookup protocol for internet applications*, IEEE/ACM Transactions on Networking, Vol. 11, No. 1, pp. 17-32, Feb. 2001.

[32] A. Rowstron and P. Druschel, *Pastry: Scalable , decentralized object location and routing for large-scale peer-to-peer systems*, IFIP/ACM International Conference on Distributed Systems Platforms, in: Lecture Notes in Computer Science, Vol. 2218, P. 329–350, 2001.

[33] B. Y. Zhao, J. Kubiatowicz, and A. D. Joseph, *Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing*, Technical Report, University of California at Berkeley Berkeley, CA, USA, 2001

[34] S. Ratnasamy, P. Francis, M. Handley, S. Shenker, and R. Karp, *A Scalable Content-Addressable Network*, In Proceedings of the ACM SIGCOMM '01 Conference, P. 161-172, 2001.

[35] M. Cai, M. Frank, J. Chen, and P. Szekely, *MAAN: A Multi-Attribute Addressable Network for Grid Information Services*, Journal of Grid Computing, Vol. 2, No. 1, P. 3-14, Mar. 2004.

[36] J. Albrecht and D. Patterson, *Scalable Wide-Area Resource Discovery*, TR CSD04-1334, University of California, 2004.

[37] S. Basu, S. Banerjee, and P. Sharma, *NodeWiz: peer-to-peer resource discovery for grids*, Proceedings of 5th IEEE International Workshop on Global and Peer-to-Peer Computing, P. 213-220, May 2005.

[38] A. R. Bharambe, M. Agrawal, and S. Seshan, *Mercury: Supporting Scalable Multi-Attribute Range Queries*, In Proc. of ACM SIGCOMM, P. 353–366, 2004.

[39] P. Rahimzadeh, M. Barati, and R. Alizadeh, *A new semantic-supported and agent-based decentralized algorithm for resource discovery in economic grid*, In Advanced Computer Theory and Engineering (ICACTE), 2010 3rd International Conference on, P. V5-441-V5-445, 2010.

[40] J. Wang, Y. Xu, G. Liu, Z. Pan, and Y. Hao, *A New Resource Discovery Mechanism with Negotiate Solution Based on Agent in Grid Environments*, The 3rd International Conference on Grid and Pervasive Computing - Workshops, P. 23-28, May 2008.

[41] S. Sotiriadis, N. Bessis, H. Ye, P. Sant, and C. Maple, *Towards decentralized grid agent models for continuous resource discovery of interoperable grid Virtual Organizations*, In Digital Information Management (ICDIM), 2010 Fifth International Conference , P. 530-535, 2010.

[42] M. Singh, X. Cheng, and R. Belavkin, *Resource Discovery Using Mobile Agents*, Fifth International Conference on Frontier of Computer Science and Technology, P. 72-77, Aug. 2010.

[43] S. M. Fattahi and N. M. Charkari, *Distributed Resource Discovery in Grid with Efficient Range Query*, In Computer Conference, CSICC 2009. 14th International CSI, no. Tehran, P. 335-340, 2009.

[44] L. R. Kinder et al., *Emergence of Scaling in Random Networks*, Science, Vol. 286, No. 5439, P. 509-512, October 1999.

[45] A. Kovvur, R.M.R. Kadappa, V. Ramachandram, S. Govardhan, *Adaptive Resource Discovery Models and Resource Selection in Grids*, In Parallel Distributed and Grid Computing (PDGC), 1st International Conference on, P. 95-100, 2010.

[46] T. Kocak and D. Lacks, *Grid Resource Discovery over Distributed Routers*, International Conference In Information Science and Applications, P. 1-8, 2010.

[47] R.-S. Chang and M.-S. Hu, A *resource discovery tree using bitmap for grids*, Future Generation Computer Systems, Vol. 26, No. 1, P. 29-37, Jan. 2010.

[48] L. M. Khanli and S. Kargar, *FRDT: Footprint Resource Discovery Tree for grids*, Future Generation Computer Systems, Vol. 27, No. 2, P. 148-156, Feb. 2011.

[49] A. Gallardo, L. Diaz de Cerio, and K. Sanjeevan, *HGRID: A self configuring Grid Resource Discovery*, International Conference on Complex, Intelligent and Software Intensive Systems, P. 833-838, Jun. 2008.

[50] H. Ren, Z. Wang, and Z. Liu, *A Hyper-cube based P2P Information Service for Data Grid*, In GCC '06 Proceedings of the Fifth International Conference on Grid and Cooperative Computing, IEEE Computer Society Washington, DC, USA, Vol. 973, No. 60573135, P. 508-513, 2006.

[51] D. D. H. Miriam and K.S.Easwarakumar, *An Efficient Resource Discovery Methodology for HPGRID Systems*, International Journal of Advanced Computer Science and Applications(IJACSA), Vol. 2, No. 1, P. 59-68, 2011.

[52] M. H. Khoobkar and M. Mahdavi, *Enabling Efficient Peer to Peer Resource Discovery in Dynamic Grids Using Variable Size Routing Indexes*, 10th International Symposium on Pervasive Systems, Algorithms, and Networks, P. 691-695, 2009.

[53] M. Marzolla, M. Mordacchini, and S. Orlando, *Peer-to-peer systems for discovering resources in a dynamic grid*, Parallel Computing, Vol. 33, No. 4-5, P. 339-358, May 2007.

[54] A. . Vashisht, P. Sharma, *Decentralized P2P Grid Resources Discovery Model in LC-Trie Structured Overlay*, In Parallel Distributed and Grid Computing (PDGC), 2010 1st International, P. 330-333, 2010.

[55] J. Fu, O. Hagsand, and G. Karlsson, *Improving and Analyzing LC-Trie Performance for IP-Address Lookup*, Journal of Networks, Vol. 2, No. 3, P. 18-27, Jun. 2007.

[56] S. Janson and W. Szpankowski, *Partial fillup and search time in LC tries*, ACM Transactions on Algorithms (TALG), Vol. 3, No. 4, P. 44, Nov. 2007.

[57] M. Z. C.Hu, *A Grid Resource Discovery Method Based on Center-distributed Virtual Community*, In Fourth International Conference on Genetic and Evolutionary Computing, P. 406-409, 2010.

[58] X. Sun, Z. Guo, C. Science, and H. Mechanic, *A Resource Discovery Mechanism Integrating P2P and Grid*, In Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE, P. 336-339, 2010.

[59] G. Pipan, *Use of the TRIPOD overlay network for resource discovery*, Future Generation Computer Systems, Vol. 26, No. 8, P. 1257-1270, Oct. 2010.

[60] Y.-H. Lin, W.-C. Chung, K.-C. Lai, K.-C. Li, and Y.-C. Chung, *SARIDS: A Self-Adaptive Resource Index and Discovery System*, 10th International Symposium on Pervasive Systems, Algorithms, and Networks, P. 521-526, 2009.