A SCALABLE ARCHITECTURE FOR DISCOVERY AND COMPOSITION IN P2P SERVICE NETWORKS

Agostino Forestiero, Carlo Mastroianni
CNR-ICAR, Rende (CS), Italy
{forestiero,mastroianni}@icar.cnr.it

Harris Papadakis, Paraskevi Fragopoulou*
Institute of Computer Science, Foundation for Research and Technology-Hellas
P.O. Box 1385, 71 110 Heraklion-Crete, Greece
{fragopou,adanar}@ics.forth.gr

Alberto Troisi, Eugenio Zimeo
Department of Engineering, University of Sannio, Benevento, Italy
{altroisi,zimeo}@unisannio.it

Abstract The desirable global scalability of Grid systems has steered the research towards the employment of the peer-to-peer (P2P) paradigm for the development of new resource discovery systems. As Grid systems mature, the requirements for such a mechanism have grown from simply locating the desired service to compose more than one service to achieve a goal. In Semantic Grid, resource discovery systems should also be able to automatically construct any desired service if it is not already present in the system, by using other, already existing services. In this paper, we present a novel system for the automatic discovery and composition of services, based on the P2P paradigm, having in mind (but not limited to) a Grid environment for the application. The paper improves composition and discovery by exploiting a novel network partitioning scheme for the decoupling of services that belong to different domains and an ant-inspired algorithm that places co-used services in neighbouring peers.

Keywords: Peer to peer, Service composition, Partitions, Ant-algorithm

*With the Department of Applied Informatics and Multimedia, Technological Educational Institute of Crete, Greece.

1. Introduction

Future applications and services in Pervasive and Grid environments will need to support more and more distributed and collaborative processes. So, systems should have the ability to cope with highly dynamic environments in which resources change continuously and in unpredictable ways, and might even be not existent when designing the system, thus calling for runtime discovery and composition.

While expectations on the quality of these systems are increasing dramatically, current methods, techniques, and technologies are not sufficient to deal with adaptive software in such dynamic environments.

Service-oriented architecture (SOA) represents a promising model for these environments. Services, in fact, are becoming important building blocks of many distributed applications where a loose connection among components represents a key aspect to better implement functional distribution (i.e. contexts in which distribution is fundamental for implementing applications) and scalability (i.e. the ability to easily extend functional properties of a system). SOA can be usefully exploited in Grid computing to dynamically acquire available computational, communication or storage resources characterized by desired QoS offerings, to compose high-level functions for solving complex problems or to enable virtual organizations.

Despite its application in many domains, the supervised approach of SOA-based technology for service discovery, composition, and execution can be a strong limitation since it represents a bottleneck from a performance point of view and imposes a centralized knowledge to discover services useful to address a specific goal.

The paper presents an approach based on P2P to overcome currently adopted connection and coordination models, which enables fully distributed and cooperative techniques for discovery, composition and enactment of services, optimized through semantic overlays. In particular, the paper shows a technique that reduces the time for automatic service composition with respect to the centralized approach. The technique is mapped on P2P networks by exploiting two mechanisms for improving performance and scalability: (1) network partitioning to reduce message flooding and (2) an ant-inspired algorithm that allows for an efficient reorganization of service descriptors. This way, a service discovery procedure can discover the basic components of a composite service in a shorter time and with lower network traffic.

These two techniques, respectively proposed by FORTH [6] and CNR-ICAR [3] are integrated with a previous work implemented at the University of Sannio on P2P service composition [7] to improve its performance and scalability.

The rest of the paper is organized as follows. Section 2 analyzes the state of the art in service discovery and composition by discussing related work.

Section 3 presents a distributed planning technique to compose services in a workflow by exploiting a P2P model and discusses some possible mapping on P2P networks. Section 4 proposes a first improvement at network level by transforming an unstructured P2P network in a semi-structured one based on partitions that capture domains of knowledge in the service network. Section 5 shows an ant-inspired algorithm to aggregate descriptors of services related to the solution of the same problem or the most used services for solving a specific problem. Finally, Section 6 concludes the paper and highlights future work among the CoreGrid partners.

2. Related Work

Service discovery is typically performed through centralized middleware components, such as registries, discovery engines and brokers. They will become a serious bottleneck when the number of services and organizations using services will grow, since service discovery requires using sophisticated matchmaking algorithms to identify semantic similarities between consumers template and providers target descriptions.

Distributed registries and registry federations have been proposed as a first approach to avoid bottlenecks in a service network [8–9]. METEOR-S [10] and PYRAMID-S [11], on the other hand, propose a scalable P2P infrastructure to federate UDDI registries used to publish and discover services; it uses an ontology-based approach to organize registries according to a semantic classification based on the kind of domains they serve. Even though these solutions well address scalability and fault-tolerance through federation of registries, they use a structured topology of registries built on the basis of a domain-specific ontology, so enforcing significant constraints on publication policies.

In [12], the authors propose a distributed federation of registries coordinated by a publish/subscribe infrastructure that is able to dispatch to the interested clients service availability as soon as a service is published in the network, by adopting a bus-oriented infrastructure to decouple registries. The proposed architecture is flexible but scalability is constrained by the number of organizations connected to the bus. Anyway, the infrastructure is mainly adopted to discover and not to compose existing services to solve a more complex problem than the ones directly solved with atomic published services.

Here, we propose a P2P model to exploit cooperative approaches for service discovery, composition and enactment. This goes beyond existing P2P technologies such as Gnutella and JXTA, which implement a primitive form of service composition through the dynamic construction of paths through the network to address queries.

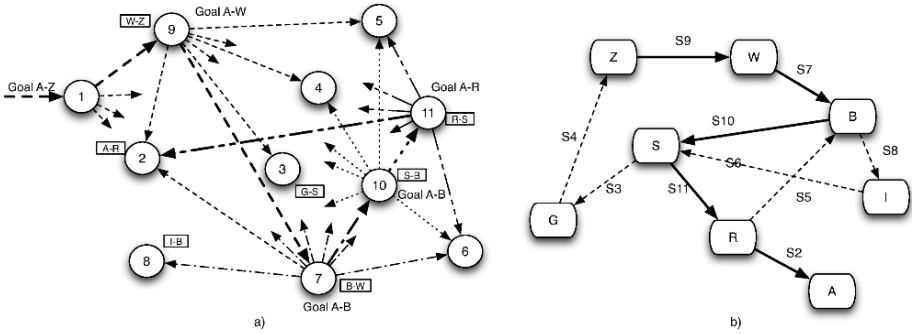

Figure 1. Cooperative composition of services to satisfy a goal. a) Space of peers and services. b) Space of states

3. Self-organizing P2P Service Network

A service network should be able to self-organize in a dynamic and adaptive way, in order to follow environmental changes and to structure the knowledge for continually optimizing service discovery and composition. In such a system of services, nodes should be able to communicate to find each other through discovery mechanisms that ensure high efficiency and scalability, with the aim of reducing response times. To this end, each node should be able to interpret an incoming goal and to give a partial or total contribution to the solution, even individuating other nodes in the network able to contribute with a piece of knowledge.

Figure 1 a) shows an example of cooperative composition in the space of services. Each node represents a peer hosting one or more services. Each sevice published on a peer exposes one operation identified through the label $Pr \rightarrow Po$, which means that Pr is the precondition and Po is the postcondition of an operation. Node 1 injects in the network a goal $(A \rightarrow Z)$ with the aim of discovering and composing the services whose execution changes the state of the system from A to Z.

Figure 1 b) shows the composition in the space of states. Each node represents a state whereas the transition between two states identifies a service that changes the system from a state to another one. The composition is the shortest path between the postcondition Z and the precondition A of the desired abstract services.

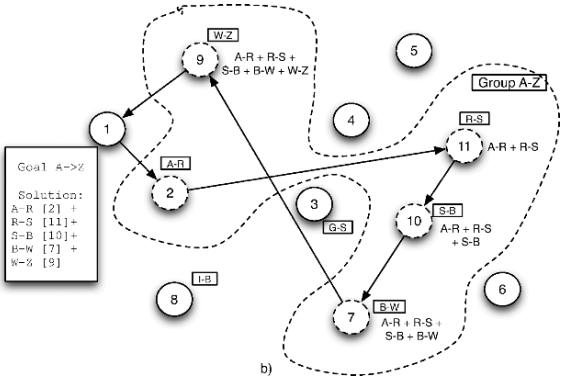

Figure 2. Group identification after a composition

3.1 Discovery and composition

Each node in the network contributes to discover the peers that can originate useful compositions. According to the P2P model, peers become a crucial part of the architecture, since with this model the network lacks of structural components for discovery and composition.

Each peer is responsible of receiving requests from other nodes (goals), and fulfilling them (i) by relying on service operations or lower level features available on each peer or (ii) by forwarding the request to other known peers (see Fig. 1 a). In many cases a peer can be able to fulfill a request by composing some of its operations with operations made available by other peers (see peers 1, 9, 7, 11). In such a case, a peer is also responsible of composing these operations, to fulfill either partially or totally the request received.

When a goal is resolved, the submitter peer receives either the composition of services or simply the identifier of the first service/peer to contact in order to start a distributed execution.

3.2 Network topology and overlays

When a composition is identified (see Fig. 2), the network implicitly aggregates the participating peers to form a new group that will simplify successive discovery and composition operations. This process is executed continually in the network, giving rise to several virtual layers of peers able to solve different problems at different abstraction layers. Therefore we can imagine having two distinct dimensions for the specialization of services: (1) a first dimension that organizes services according to the domain in which they are used, and (2) a second dimension that organizes services in groups to simplify new and more complex compositions.

Simulation experiments with compositions of 10 services belonging to a space ranging from 10 to 640 services/peers have demonstrated a speed-up ranging from 3.5 to 19 without grouping and from 4.7 to 129 with groups. This demonstrates that mapping the concepts presented above on a P2P network is a crucial concern to achieve high performance.

However, techniques based on network information limit the traffic in the network since queries are routed only to the peers that host the desired resource. In Distributed Hash Tables (DHT) for example, each peer or resource is mapped to a unique identifier in a known identifier space. The combination of unique identifiers and a known space from which these identifiers are drawn, allows routing to be achieved efficiently. The payoff for this efficiency however, is that such architectures require a highly structured network and do not well support ad hoc configurations. The strong constraints imposed to the publication of service advertisements severely limit the possibilities for cooperation among peer nodes. The lack of a single point of failure in P2P unstructured networks ensures a better fault tolerance of the overall system whereas the availability of a potentially non-limited storage capacity for indexing increases significantly network scalability. Unfortunately, unstructured networks typically adopt flooding to broadcast discovery messages to all the reachable peers connected to the network.

4. Network partitioning

One way to reduce the cost of flooding is to partition the overlay network into a small number of distinct subnetworks and to restrict the search for individual request to one network partition. The *Partitions* scheme, proposed in this section, enriches unstructured P2P systems with appropriate data location information in order to enable more scalable resource discovery, while not affecting at all the self-healing properties and the inherent robustness of these systems.

More specifically, Partitions employ a universal uniform hash function to map each keyword to an integer, from a small set of integers. Each integer defines a different category. Thus, keywords are categorized instead of services/content.

The keyword categories are exploited in a 2-tier architecture, where nodes operate as Ultrapeers and/or Leaves. The partitioning of the network is performed as follows (see Fig. 3):

- Each Ultrapeer is randomly and uniformly assigned responsibility for a single keyword category. Ultrapeers responsible for the same category form a random subnetwork. As a consequence, the network overlay is partitioned into a small number of distinct subnetworks, equal to the number of available categories.

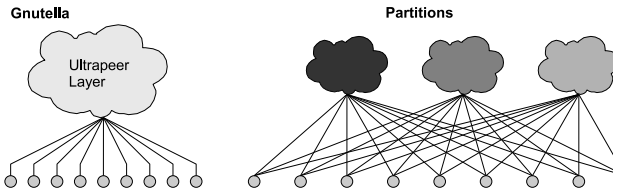

Figure 3. Illustration of the Gnutella network and the Partitions design.

- Leaves randomly connect to one Ultrapeer per subnetwork. Furthermore, each Leaf sends to each Ultrapeer it is connected to all its keywords, in the form of a bloom filter, that belong to that same category. Thus, an innovative index splitting technique is used. Instead of each Leaf sending its entire index (keywords) to each Ultrapeer it is connected to, each Leaf splits its index based on the defined categories and constructs a different bloom filter for each keyword category. Each bloom filter is then sent to the appropriate Ultrapeer. An illustration of this technique can be found in Fig. 4.

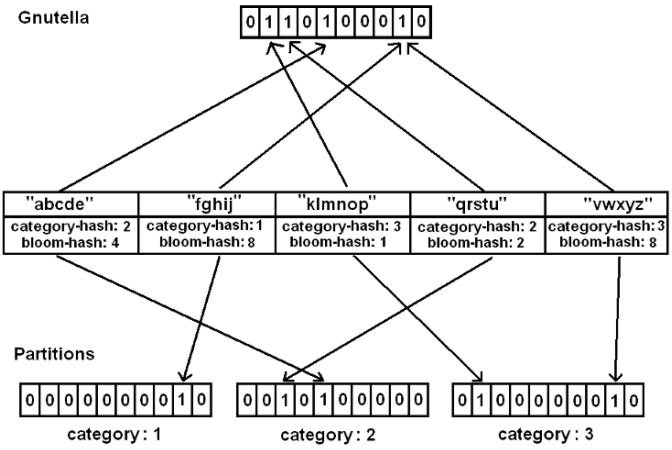

Figure 4. The Partitions and Gnutella bloom filters.

We should emphasize that in this design Ultrapeers are de-coupled from their content, meaning that peers operating as Ultrapeers will have to also operate as Leaves at the same time in order to share their own content, which spans several categories. Furthermore, even though in this design each Leaf connects to more than one Ultrapeers, the volume of information it collectively transmits

to all of them is roughly the same since each part of its index is send to a single Ultrapeer.

The Partitions scheme is demonstrated in Fig. 3. The unstructured overlay network is partitioned into distinct subnetworks, one per defined category. A search for a keyword of a certain category will only flood the appropriate subnetwork and avoid contacting Ultrapeers in any other network partition. The benefit of this is two-fold. First, it reduces the size of the search for each individual request. Secondly, it allows each Ultrapeer to use all its Ultrapeer connections to connect to other Ultrapeers in the same network partition, increasing the efficiency of 1-hop replication at the Ultrapeer level. One-hop replication dictates that each Ultrapeer contains an index of the contents of its neighbouring Ultrapeers (including the contents of their Leaves).

There are, however, two obvious drawbacks to this design. The first one is due to the fact that each Leaf connects to more than one Ultrapeers, one per content category. Even though each Leaf sends the same amount of index data to the Ultrapeers collectively upon connection as before, it requires more keepalive messages to ensure that its Ultrapeer connections are still active. Keepalive messages however are very small compared to the average Gnutella protocol message. In addition, query traffic is used to indicate liveliness most of the time, thus avoiding the need for keepalive messages.

The second drawback arises from the fact that each subnetwork contains information for a specific keyword category. Requests however may contain more than one keywords and each result should match all of them. Since each Ultrapeer is aware of all keywords of its Leaves that belong to a specific category, it may forward a request to some Leaf that contains one of the keywords but not all of them. This fact reduces the efficiency of the 1-hop replication at the Ultrapeer level and at the Ultrapeer to Leaf query propagation. This drawback is balanced in two ways. The first is that even though the filtering is performed using one keyword only, Leaves' bloom filters contain keywords of one category, which makes them more sparse, thus reducing the probability of a false positive. Furthermore, the most rare keyword can be used to direct the search, further increasing the effectiveness of the search method.

Simulation experiments have been conducted for 10, 30, and 60 network partitions. The results demonstrated in Fig. 5 show that the Partitions scheme reduces significantly the number of messages generated through flooding while simultaneously reducing the network load observed by each Ultrapeer.

5. Ant-inspired reorganization of descriptors

Inside each partition, the construction of a composite service (workflow) needs the identification of the basic services that will compose the workflow, and the discovery of such services on the network. This is generally reduced

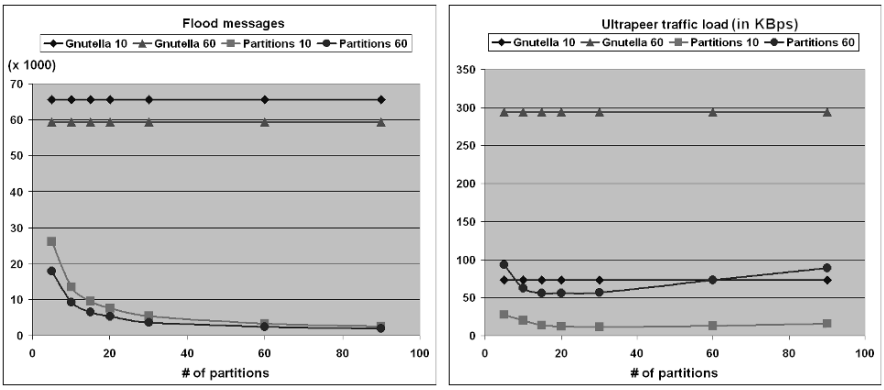

Figure 5. (a) Number of messages generated in one flood.(b) Query traffic observed by each Ultrapeer.

to the problem of finding *service descriptors*, through which it is possible to access the corresponding services.

In general, the construction of a workflow implies the generation of a discovery request for each of the required basic services, which can result in long discovery times and high network loads. The technique described in Section 4 allows for reducing flooding generated by discovery operations (by creating different domains). To further enhance performances, it would be useful to place descriptors of services that are often used together (i.e., in the same composite services) in restricted regions of the system, so that a single discovery operation will have high chances to find all or most of the required services in a short time (groups). Accordingly, we propose an ant-inspired technique to reorganize and sort the service descriptors in order to facilitate the composition of workflows.

Descriptors are indexed through bit strings, or *keys*, that are generated by a hash function. The hash function is assumed to be locality preserving [2, 5], which assures that similar descriptor keys are associated to similar services. In this context, two services are defined as *similar* if they are often used together to compose a workflow. This type of similarity must be based on a statistical analysis of co-occurrences of services in the workflows characterized by similar semantics.

Our algorithm is inspired by the behavior of some species of ants [1], that sort items or corpses within their environment. The algorithm described in this paper has been specifically designed to *sort* service descriptors, i.e., to place descriptors of services that are often co-used in composite services in neighbor peers, in order to facilitate and speed up their discovery. This work is inspired by the work of Lumer and Faieta [4], who devised a method to spatially sort data items through the operations of robots. In our case, descriptors are not only

sorted, but also *replicated*, in order to disseminate useful information on the system and facilitate discovery requests. The behaviour of ants is here imitated by mobile agents that, while hopping from one peer to another, can copy and move service descriptors. Agents are able to disseminate and sort descriptors on the basis of their keys. Therefore, services descriptors individuated by similar keys will likely be placed in neighbour peers. This facilitates the composition of workflows in three ways:

(i) the ant-inspired agents are able to create and disseminate replicas of service descriptors, thus giving discovery operations more chances to succeed and improving the fault tolerance characteristics of the system;

(ii) the discovery of a single service is facilitated because discovery messages can be driven towards the target descriptor in a very simple way. At each step the discovery message is sent to the neighbor peer whose descriptors are the most similar to the target descriptor. Due to the sorting of descriptors in most cases this method allows queries to reach peers that store a significant number of useful descriptors;

(iii) once a target descriptor has been reached, it is possible to locate other services, needed in the same workflow, in the same network region. Indeed, since services that are often co-used have similar keys, they have likely been placed into very close peers by mobile agents.

The ant-inspired algorithm is briefly described in the following, but more details can be found on [3]. Periodically each agent performs a small number of hops among peers. When an agent arrives at a new peer, if it is carrying some descriptors it must decide whether or not to *drop* these descriptors whereas, if it is currently unloaded, it must decide whether or not to *pick* one or more descriptors from the local host. When performing a pick operation, an agent must also decide if the descriptor should be replicated or not. In the first case, the descriptor is left on the current peer and the agent will carry a replica; in the other case, the descriptor is taken from the peer and carried by the agent. This way, agents are able to replicate, move and reorganize the descriptors.

In both cases, agent decisions are based on a similarity function, f, reported in formula (1), which is based on the basic ant algorithm introduced in [4]. This function measures the average similarity of a given descriptor $\overline{d}$ with all the descriptors d located in the local region R. In formula (1), N_d is the overall number of descriptors maintained in the region R, while $H(d, \bar{d})$ is the Hamming distance between d and $\overline{d}$. The parameter α is set to $B/2$, which is half the value of the maximum Hamming distance between vectors having B bits. The value of f assumes values ranging between -1 and 1, but negative values are truncated to 0.

$$f(\bar{d}, R) = \frac{1}{N_d} \cdot \sum_{d \epsilon R} (1 - \frac{H(d, \bar{d})}{\alpha}) \qquad (1)$$

The probability of picking a descriptor stored in a peer must be inversely proportional to the similarity function f, thus obtaining the effect of averting a descriptor from co-located dissimilar descriptors. Conversely, the probability of dropping a descriptor carried by an agent must be directly proportional to the similarity function f, thus facilitating the accumulation of similar descriptors in the same local region.

The pick and drop probability functions, *Ppick* and *Pdrop*, are defined in formulas (2)a and (2)b. In this functions, the parameters k_p and k_d, whose values are comprised between 0 and 1, can be tuned to modulate the degree of similarity among descriptors.

$$_{(a)}\ Ppick = \left(\frac{k_p}{k_p + f}\right)^2 \quad _{(b)}\ Pdrop = \left(\frac{f}{k_d + f}\right)^2 \tag{2}$$

After evaluating the pick or drop probability function (which function depends whether the agent is carrying descriptors or not), the agent computes a random number comprised between 0 and 1 and, if this number is lower than the value of the corresponding function, it executes the pick or drop operation for the descriptor under examination. The inverse and direct proportionality with respect to the similarity function f assures that, as soon as the possible initial equilibrium is broken (i.e., descriptors having different keys begin to be accumulated in different Grid regions), the reorganization of descriptors is more and more facilitated.

6. Conclusions

The paper proposes an innovative approach to the distributed and cooperative composition of Grid (and not only) services based on: (1) AI planning techniques, (2) service grouping and (3) overlays in large P2P networks. To this end, we integrated a system for distributed service composition with (1) an intelligent, ant-inspired search mechanism able to self-organize the location of the services to facilitate discovery and (2) a partitioning technique that reduces flooding. The end system is a composite mechanism that can scale to a large number of services and peers. Future work will consolidate the mechanism in a P2P infrastructure with the aim of comparing the results with those obtained by simulations. Furthermore, new techniques will be exploited for identifying service similarities for composition and their impacts on network partitioning.

Acknowledgments

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

References

[1] Eric Bonabeau, Marco Dorigo, and Guy Theraulaz. *Swarm intelligence: from natural to artificial systems*. Oxford University Press, New York, NY, USA, 1999.

[2] Min Cai, Martin Frank, Jinbo Chen, and Pedro Szekely. Maan: A multi-attribute addressable network for grid information services. In *GRID '03: Proceedings of the Fourth International Workshop on Grid Computing*, page 184, Washington, DC, USA, 2003. IEEE Computer Society.

[3] Agostino Forestiero, Carlo Mastroianni, and Giandomenico Spezzano. Antares: an ant-inspired p2p information system for a self-structured grid. In *BIONETICS 2007 - 2nd International Conference on Bio-Inspired Models of Network, Information, and Computing Systems*, Budapest, Hungary, December 2007.

[4] Erik D. Lumer and Baldo Faieta. Diversity and adaptation in populations of clustering ants. In *Proc. of SAB94, 3rd international conference on Simulation of adaptive behavior: from animals to animats 3*, pages 501–508, Cambridge, MA, USA, 1994. MIT Press.

[5] D. Oppenheimer, J. Albrecht, D. Patterson, and A. Vahdat. Design and implementation tradeoffs for wide-area resource discovery. In *Proc. of the 14th IEEE International Symposium on High Performance Distributed Computing HPDC 2005*, Research Triangle Park, NC, USA, July 2005.

[6] Harris Papadakis, Paraskevi Fragopoulou, Marios Dikaiakos, Alexandros Labrinidis and Evangelos Markatos. *Divide Et Impera: Partitioning Unstructured Peer-to-Peer Systems to Improve Resource Location.* CoreGRID Springer Volume, 2007.

[7] Alberto Troisi, Eugenio Zimeo Self-Organizing Service Network in a P2P environment. *Technical Report*, Research Centre on Software Technology - University of Sannio, Italy, 2007.

[8] ebXML. ebXML: electronic business using extensible markup language. http://www.ebxml.org

[9] UDDI 3.0 Universal description, discovery and integration version 3. http://www.uddi.org

[10] Kunal Verma, Kaarthik Sivashanmugam, Amit Sheth, Abhijit Patil, Swapna Oundhakar, John Miller. *METEORĐS WSDI: A Scalable P2P Infrastructure of Registries for Semantic Publication and Discovery of Web Services.* Information Technology Management, (6)1:17-39, 2005.

[11] T.Pilioura, G. Kapos, and A. Tsalgatidou. PYRAMID-S: a scalable infrastructure for semantic web services publication and discovery. In *Proc. of the 14th International Workshop on Research Issues on Data Engineering*, 28-29 March 2004.

[12] Luciano Baresi, Matteo Miraz. A Distributed Approach for the Federation of Heterogeneous Registries. In *Proc. of ICSOC 2006*, Chicago, USA, 2006.