

# PYRAMID-S: A Scalable Infrastructure for Semantic Web Service Publication and Discovery

Thomi Pilioura<sup>1</sup>  
University of Athens, Dept. of Informatics,  
Panepistimiopolis, TYPA Buildings  
Ilisia, 157 84, Athens, Greece  
{thomi, atsalga}@di.uoa.gr

\*Georgios-Dimitrios Kapos

Aphrodite Tsalgatidou  
\*Harokopio University of Athens, NOC  
70 El. Venizelou Str.  
176 71, Athens, Greece  
gdkapos@hua.gr

## Abstract

*Web services enhance current web functionality by altering its nature from document to service-oriented. As the number of web services increases, it becomes increasingly important to provide a scalable infrastructure of Registries that allows both developers and end-users to perform discovery of semantic web enabled services. The discovery of services needs to be based on QoS characteristics in order to enable result ranking and service selection. Current web service publication and discovery mechanisms, such as UDDI, either address these issues partially or not at all. In this paper, we build on the enabling technologies of Web Services, Peer-to-Peer and Semantic Web in an attempt to address all these important dimensions of service publication and discovery. More specifically, we use a hybrid peer-to-peer topology to organize Registries based on domains. In such a model, each Registry retains its autonomy, meaning that it can use the publication and discovery mechanisms as well as the ontology of its choice.*

## 1. Introduction

The web service (WS) paradigm is transforming the Web from a provider of static pages to a provider of interactive, automated and intelligent services that interact via the Internet [12][13]. Multiple web services may interoperate to perform tasks, provide information, transact business and generally take action for users, dynamically and on demand. The WS model simplifies business application development and interoperation, as it entails code reuse and loose coupling between services thanks to the adoption of widely accepted standards. Additionally, it may serve end-user needs by providing an intuitive, browser-based interface that enables users to choose, configure and assemble their own web services.

However, in order to employ its full potential, the WS paradigm must be supported by an appropriate service publication and discovery infrastructure. At present, the most prevalent standard for WS publication and discovery is the UDDI [18] specification. However, the effectiveness of UDDI is limited as it is characterized by a number of shortcomings:

- **Deficient service description:** UDDI is mainly used in combination with WSDL [17]. WSDL is an XML grammar for specifying functional and syntactic aspects of a web service such as what it does, where it is located and how it is invoked. The only non-functional and semantic information about services is provided by using the various standard UDDI taxonomies (related industry, products or services offered and geographical region).
- **Deficient description of requestor's needs:** There is no formal way of expressing requestor's needs. Instead, a service requestor retrieves advertisements out of the UDDI Registry based on keyword search (exact pattern matching) on some fields such as name, identifier or taxonomy. The latter is the only field conveying semantic information as it enables users to search the Registry by industry, product category or geographic location. However, this is not enough for achieving automated discovery as two identical service Registrations in the UDDI could mean totally different things, depending on the context in which they are used. Thus UDDI only provides a first level filter in the discovery process. Further filtering is done by manual inspection of the service descriptions.
- **Lack of end-user friendly interface:** Searching functionality in UDDI is primarily addressed to developers or programs. It does not make any provision for the end-users who are the final recipients of the WS functionality.
- **Scalability issue:** The centralized indexing scheme provided by UDDI does not scale well because the number and physical distribution of the UDDI clients

<sup>1</sup> Ms. Thomi Pilioura is also with the National Bank of Greece

can quickly overwhelm this centralized configuration and can lead to serious performance bottlenecks. Adding more servers or implementing load-balancing strategies does not constitute a practical solution as it implies high cost for the operators of the UDDI nodes.

In this paper we propose an infrastructure for web service publication and discovery (PYRAMID-S hereinafter), which addresses the above limitations by combining the technologies of Web Services, Semantic Web and Peer-to-Peer Networking. More specifically, the main contributions of this infrastructure are the following:

- Web service publication and discovery based on syntactic as well as semantic information
- Use of Quality of Service (QoS) characteristics in publication and discovery in order to enable result ranking and service selection
- Use of a scalable infrastructure which organizes Registries based on domains and enables users to access multiple Registries through a unified view
- Preservation of the autonomy of Registries by enabling the accommodation of different publication and discovery mechanisms
- Web service discovery for both end-users and developers

The rest of the paper is organized as follows: Section 2 presents a scenario illustrating the publication and discovery of services and highlights the challenges involved. Section 3 presents the PYRAMID-S infrastructure. The PYRAMID-S protocols are described in Section 4. Section 5 presents related work. Finally, we give our conclusions and we outline our future work plans.

## 2. Motivation

Consider a bank with a large number of departments, each having lots of web services. The web services of each department may be used by the employees and developers of the same or other departments, the employees and developers of the business partners of the bank as well as by the Internet customers of the bank. The use of the services presupposes a means for making them known to others (service publication) and for finding and selecting the appropriate services (service discovery and selection).

If such a means is available, then the developer can use simple web services in his/her programs or compose services when the offered services do not satisfy his/her needs. For example, consider the case where the developer wants to implement a loan approval service. This can be achieved by composing various services such as:

- **CreditScoreCalculation:** A web service that calculates the applicant's credit score based on the application form data (e.g. type of loan, applicant's annual revenue and age) and the credit score card used by the financial institution.
- **CreditProfileCheck:** This web service interacts with a Credit Profile Databank system in order to check if the applicant is solvent or not.

From the end-user's point of view, s/he needs an infrastructure that enables him/her to:

- search for a loan approval service where the process time for approval is less than 30 minutes
- search for a loan approval service provided by National Bank of Greece that costs less than 10 euro

The above scenario highlights the following challenges:

- **Challenge 1: Publication and Discovery based on syntactic and semantic service characteristics.** The syntactic information comprises the service name and a short textual service description. The semantic information attributes machine-understandable meaning to the concepts of the service description.
- **Challenge 2: Publication and discovery enriched with QoS characteristics. (e.g. cost, response time) characteristics.** The QoS metrics of a service (e.g. cost, response time) should be taken into account when searching for web services. This facilitates differentiation among services with the same functional characteristics and also gives some degree of confidence to the WS requestors about the quality of the service they are going to invoke.
- **Challenge 3: Scalable publication and discovery.** A scalable infrastructure can be used with each department running a department specific Registry and each Registry conforming to a department specific ontology. The aim is to shift hardware, support and upkeep resources from the burden of few Registry operators to the responsibility of a great number of Registry operators which may also be the service providers. This results to better and up-to-date content.
- **Challenge 4: Autonomy of Registries.** The lack of standards for the semantic annotation of web services is still a major shortcoming of current Web technology. Thus, the infrastructure should accommodate different publication and discovery mechanisms. However, the physical distribution of the registries should be transparent to the users so that they can view them as a single collection accompanied with uniform tools for search, retrieval and display of information.
- **Challenge 5: Web service discovery for both end-users and developers.** An intuitive interface must be provided that will enable the users to specify their needs in an easy way. The user interface should also

enable the end-users to directly invoke the selected service by allowing him/her to enter the values of the input parameters of the service. This would also be useful for developers who wish to test the service before integrating it in their applications.

### 3. The PYRAMID-S infrastructure

In order to meet the challenges listed above we propose the PYRAMID-S infrastructure. The challenge of semantics is addressed by using a number of ontologies which are presented in the following section. Then we present the general architecture of the proposed system and the functionalities provided by each layer of the architecture.

#### 3.1. PYRAMID-S ontologies

In PYRAMID-S we address the problem of semantics by using a number of ontologies. Four different types of ontologies are used:

**Standard Domain Ontology (SDO):** This ontology contains concepts that are likely to be used by WS in a particular domain. For example a SDO for the Loans Services domain may include concepts such as LoanAmount, ServiceFee, InterestRate and Term. The SDO of a specific domain is the default ontology of the PYRAMID-S infrastructure for that domain. The Registries conforming to that SDO use this ontology for the semantic publication and querying of the web services they hold.

**Registry Domain Ontology (RDO):** Registries may either adopt the SDO or use their own domain ontology (RDO). In the second case the Registry has to provide a mapping from its own ontology to the SDO.

**Domain Classification Ontology (DCO):** This ontology maintains relationships among domains and mappings of each Registry of the PYRAMID-S infrastructure to one or more domains. In addition, it stores properties of Registries, such as the access URL, an indication of whether it supports publication and discovery enriched with QoS characteristics, the Registry provider details, the access URL of the RDO used (in case of non-conformance to SDO) and the constraints in accessing that Registry. Figure 1 shows a sample DCO in the form of a tree. Information regarding the registries in a DCO may be expressed as a set of tuples  $T_i = \langle R_i, D_i, A_i \rangle$ , where  $R_i$  is the access URL of a Registry,  $D_i$  is a domain, and  $A_i$  are the properties of the Registry. The tuples are identified by the combination of  $R_i$  and  $D_i$ , that is for any  $x, y$  ( $T_x \neq T_y \Rightarrow R_x \neq R_y \vee D_x \neq D_y$ ).

**QoS ontology:** Based on previous studies [10][11][14] and our experience in the WS area, we have constructed a QoS ontology composed of the following dimensions: response time, cost, availability and security. This

ontology is used in combination with SDO for the semantic publication and querying of the web services. We assume that this ontology is commonly accepted and used by all Registries.

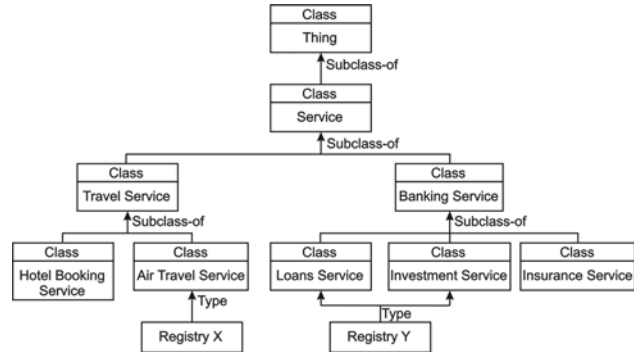


Figure 1 A sample DCO

#### 3.2 The PYRAMID-S architecture

The main motivation of our work is to provide a scalable architecture for service publication and discovery. This can be achieved by distributing services in domain specific Registries. This enables more pertinent discovery of services as the selection of a Registry containing services of a particular domain works as a first filter in the discovery process. For example, if a Registry is related to the Financial Services domain, it will maintain web services specific to that domain and queries for web services in the Financial Services domain can be routed to it.

On the basis of the above considerations we propose the PYRAMID-S infrastructure, which consists of three layers (depicted in the right part of Figure 2):

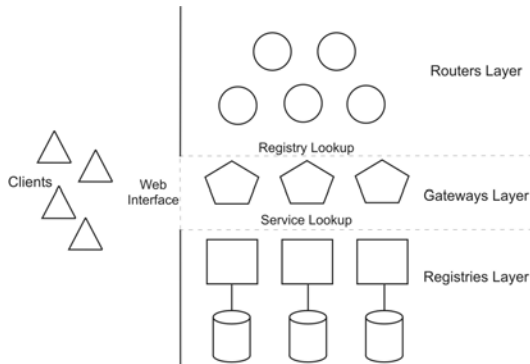
**Registries layer:** This layer consists of a number of Registries. PYRAMID-S accommodates any kind of web service Registry/Repository such as DAML-S [7][1] based registries, UDDI registries and ebXML Registries [4].

**Gateways layer:** The servers of this layer function as entry points to the PYRAMID-S system. There are several such servers that are known a priori to the clients.

**Routers layer:** This layer consists of a number of servers that provide routing service to the Gateway peers in order to forward the queries/advertisements entered in the Gateways to the appropriate domain registries.

PYRAMID-S is based on a peer-to-peer network, which provides the infrastructure for the distributed nodes of PYRAMID-S to communicate with each other (peer identification, peer discovery, etc.). The peer-to-peer topology renders PYRAMID-S scalable because it allows Routers to easily advertise and unadvertise themselves to the Gateways. More importantly, this topology ensures that there is not a single point of failure in the Routers layer.

Each node, depending on the layer it belongs, plays a particular role in the peer-to-peer network. Each peer of the Routers layer plays a role similar to that of an index server in a semi-centralized peer-to-peer network where the various peers communicate with the index server in order to obtain a reference to the data/processing that is available on the network. This implies a hybrid peer-to-peer network. In the following we present in detail the functionalities provided by each layer of the PYRAMID-S infrastructure.



**Figure 2 The PYRAMID-S architecture**

**3.2.1. Gateways layer.** As we have already mentioned the peers of this layer function as entry points to the PYRAMID-S infrastructure. A client (service requestor or service provider) directs the query/advertisement to any peer of this layer, which then redirects the query/advertisement to the appropriate Registry/Registries. The selection of the appropriate Registry/Registries is performed through the use of the Routers layer.

A Gateway provides the following functionality:

- It offers appropriate interfaces to the users for:
  - updating the DCO (adding/renaming a domain, associating a Registry to one or more domains, updating the properties of a Registry). Registry operators can categorize their Registry to more than one domain and to any level of the DCO tree.
  - entering/removing a Service Advertisement (SA). The SA consists of two parts, the Service Advertisement Template (SAT) and the WSDL document (we assume that all type of Registries use the WSDL as a basis for service description as it is the most widely accepted standard)
 

Thus SA = WSDL + SAT  
 where SAT = <SP, OPs, Os, Is, QoS>  
 (SP: service provider information, OPs: set of service operations specified with concepts from DCO, Is/Os: set of input/output parameters specified with concepts from DCO, QoS: quality of service metrics)

The use of SA ensures a uniform view for all type of Registries.

- entering a service request. The service discovery is achieved by submitting a Service Query (SQ) to a Gateway. The SQ specifies the requirements about the service to discover.

SQ = <SN, SD, SP, OP, Os, Is, QoS>

(SN: service name, SD: service textual description, SP: service provider information, OP: requested operation specified with concepts from DCO, Is/Os: set of input/output parameters specified with concepts from DCO, QoS: quality of service metrics)

Each Gateway peer provides a mediator component for each type of Registry. This component translates the SA or SQ to the Registry specific format and performs the service publication or discovery respectively. This translation also takes into account the SDO to RDO mapping in case of non-conformance to the SDO.

The elements of SAT and SQ are optional. Furthermore, some of the information included in SAT or SQ will not be used if the Registry where the publication/query takes place does not support it.

- It presents to the users the results of a query. These results are ranked based on the QoS characteristics of the discovered services. If an individual Registry is unavailable (due to network or server failure) or overloaded (resulting in a time-out) the user is alerted that results could not be returned from that Registry.
- If the user is a business user or a developer who wishes to test a web service, s/he is provided with an appropriate interface for invoking the service.
- It offers to Registry operators a mapping tool that facilitates the appropriate mapping from the Registry Domain Ontology to the Standard Domain Ontology. The resulting mapping is stored in the Routers layer.

**3.2.2. Routers layer.** This layer consists of a number of peers that provide routing service to the Gateways peers in order to forward the queries/advertisements entered in the Gateways to the appropriate domain Registries. The peers of this layer ensure high availability, protection and consistency of routing information, that is information regarding registries and their mapping to domains. If any Router is disconnected from the network, the routing service is not affected unless this peer is the last Router of the peer network. The Routers also hold the Standard Domain Ontologies (SDOs), the QoS Ontology as well as the relationship between domains and SDOs. They also hold the mapping from RDO to SDO. This mapping is necessary as no global enforcement on the use of ontologies is possible in highly autonomous environments.

**3.2.3. Registries layer.** This layer consists of a number of Registry peers that are responsible for getting the service advertisement (SA) or the service request (SQ) from the Gateway and for performing the necessary actions based on the type of the Registry and the conformity or not to the SDO.

## 4. The PYRAMID-S protocols

### 4.1. Registry insert/delete/update protocol

A Registry operator wishing to insert/update or delete a Registry in the DCO uses the appropriate interface provided by a Gateway. The Gateway contacts a Router in order to acquire the DCO and then presents the latter to the user in order to associate his/her Registry to the appropriate domain. In the case of insertion or update the Registry operator also enters or updates the properties of the Registry. If a mapping from RDO to SDO is needed, the Gateway provides the user with the appropriate interface. In this interface, the RDO and SDO (retrieved from a Router) are represented as taxonomy of concepts in a tree structure.

The user input is translated into one of the following operations on the DCO:

- Insert( $T_x$ ): Registry  $R_x$  is stated to be related to domain  $D_x$  and to provide its services with  $A_x$  properties. This operation is valid only if there is no  $T_y \in \text{DCO}$ :  $R_y = R_x \wedge D_y = D_x$ . After the operation, DCO is  $\text{DCO} + \{T_x\}$ .
- Delete( $T_x$ ): Registry  $R_x$  is no longer related to domain  $D_x$ . This operation is valid only if  $T_x \in \text{DCO}$ . After the operation DCO is  $\text{DCO} - \{T_x\}$ .
- Update( $T_x, A_x'$ ): The properties of Registry  $R_x$  for domain  $D_x$  are updated to  $A_x'$ . This operation is valid only if  $T_x \in \text{DCO}$ . After the operation DCO is  $\text{DCO} - \{T_x\} + \{(R_x, D_x, A_x')\}$ .

Insertions, deletions and updates of the DCO may be performed by any of the Routers and are propagated to all other Routers. Therefore consistency of the DCO should be assured during conflicting operations, performed by the same or distinct Routers within a short timeframe (i.e. the time needed for an operation to be propagated to the whole peer group). From the description of the DCO and of the above operations, it is clear that there are two cases of having conflicting operations:

- when two or more Routers attempt to rename the same domain of the DCO
- when insertions, deletions and updates are performed on the same couple of  $R_x$  and  $D_x$ , i.e. on the same  $T_x$

Our conflict resolution mechanism uses a request/reply/notify scheme incorporated by the Routers. This may be seen as a distributed lock management scheme, where each requesting Router requests

permission, by the Routers peer group, to perform an operation on a specific part of the DCO. At most one Router will succeed, since overall acceptance is decided upon the balance of positive and negative replies each requesting Router has received. In case the conflicting Routers receive equal positive replies, none of them receives the permission. For example, consider a network of 12 Routers. If a Router requests permission to perform an operation and receives 8 positive replies, permission to perform the requested operation is implicitly granted to it.

The scheme is implemented as follows. Whenever a Router intends to perform an operation on the DCO it sends an appropriate request to all other Routers (implicitly asking to lock a specific part of the DCO in order to perform the requested operation) and waits for their replies. Each Router maintains a log for operation requests it has approved and upon reception of a new request by another peer it replies negatively if this request conflicts with any of the requests currently present in its log. Otherwise it adds this request to its log and replies positively. Each requesting Router may infer from its log whether it has been granted the lock or not, after the reception of all replies. It is clear that only one Router will get the majority of positive replies and will therefore be granted the lock. This Router can then proceed with the requested operation by notifying the opposite minority about its prevalence and receiving positive replies by these Routers, too. All other conflicting, requested operations are cancelled by all Routers. After the conflict resolution, the winner Router commits to all other Router peers its operation, and the latter is performed by all the other Router peers, maintaining in this way consistency of all the DCOs stored in the various Routers. After having been notified by all other peers that they have performed the operation, the winner Router performs it too. The relevant log entry is also removed by all Routers.

The consistency maintenance scheme used by the Routers group presents two main advantages. First, only specific parts of the DCO are locked, allowing non-conflicting updates to be performed concurrently and therefore avoiding unnecessary delays. Second, there is no single point of failure for lock management, since it is achieved by a fully decentralized process involving the whole Routers group. The introduced communication overhead is negligible, in comparison to other communication and processing delays.

### 4.2. Publication protocol

A service provider wishes to register a service in the PYRAMID-S system. If s/he has her/his own Registry, s/he registers the service within it. If s/he does not have a Registry, s/he searches for a desirable Registry or Registries using the appropriate interface provided by one of the Gateways. The Registry can be found using various

criteria, such as the Registry Provider, the domain or combination of them. Then s/he selects the desired Registries from a list of Registries conforming to her/his criteria. The Gateway provides to the user an interface for entering the service advertisement (SA), which is mapped to the appropriate description format of each Registry (SA') by the respective mediator component. The SA' is then forwarded to the Registries conforming to the service provider's criteria. Finally each Registry informs the service provider through the Gateway for the result of his/her action.

### 4.3. Discovery protocol

A service requestor may search for a service either in a specific Registry or in all the Registries belonging to a specific domain. This is accomplished through the appropriate interface provided by a Gateway. This user interface enables the service requestor to specify the SQ describing his/her needs. The SQ is built by using concepts of the SDO which is retrieved from a Router. The service requestor may also specify relative weights corresponding to the QoS characteristics. The SQ is then mapped to the appropriate queries supported by the selected Registries which are then routed to them. Each Registry uses its own matchmaking algorithms. The results of the various Registries are returned to the Gateway which consolidates and ranks the results based on the user's predefined weights.

## 5. Related work

There are several research activities related to our work. Each of these activities addresses some of the challenges set forth in section 2. However, none of them addresses all the issues as we do in our proposed system. In the following we review the most relevant work for each of the challenges.

### 5.1. Publication and discovery based on syntactic and semantic service characteristics

DAML-S [3] is based on DAML+OIL and provides an ontology markup language expressive enough to semantically represent capabilities of WS. DAML-S also suggests using a WSDL file along with a DAML-S description to represent a service [2]. Paolucci et al [8] propose a DAML-S based matchmaking algorithm that uses only inputs and outputs defined in the same ontology. In [7] they extend their work to add semantic capability to UDDI by translating DAML-S description of a service to UDDI representation.

The approach described in [5] involves annotating and extending WSDL constructs with DAML+OIL ontological concepts and storing this semantic

information in UDDI. The advantage of this approach is that it fits better with existing industry standards, rather than requiring new infrastructure needed by DAML-S. In PYRAMID-S we propose an extended version of the WSDL annotations described in [5] in order to enrich the service advertisement (SA) and query (SQ) with QoS information.

In [9] the authors describe a methodology and a set of algorithms for WS discovery based on three dimensions: syntax, operational metrics and semantics.

PYRAMID-S accommodates all the above approaches as they may be adopted by the various Registry operators for implementing their publication and discovery mechanism. This is possible thanks to the use of uniform templates for service advertisements and service queries and the provision of mediators for each Registry type.

### 5.2. Publication and discovery enriched with QoS characteristics

UDDIe [20] is an extension to UDDI which supports the notion of "blue pages" to record user defined properties associated with a service – and to enable discovery of services based on these. In [15] a new WS discovery model is proposed which takes into account QoS requirements. This model introduces a new role, the certifier which verifies the QoS claims from the WS providers.

### 5.3. Scalable publication and discovery and autonomy of registries

The METEOR-S system described in [6] is closely related to ours as it addresses most of the challenges set forth in section 2. It uses an ontology-based approach to organize Registries, enabling domain based semantic classification of WS. The main differences between the two approaches are the following: (a) although both approaches use a peer-to-peer architecture, in PYRAMID-S we exclude the clients from the p2p network. The advantage of this is that we do not require service requestors and providers to download and install any software, (b) whereas in METEOR-S approach the client selects the domain ontology it prefers and sends the advertisement or query to a single Registry, our system supports publication and discovery across multiple Registries thanks to the use of RDOs and mediators, (c) whereas in METEOR-S the editing of the domain classification and registry information is performed only in a single peer resulting in a single point of failure, in PYRAMID-S we allow editing in a number of peers making provision for data consistency.

## 5.4. WS Discovery for end-users and developers

The basic functionality of the composer presented in [14] is to let the users invoke web services annotated with DAML-S. A user is presented with a list of services registered to the system and can execute an individual web service by entering the values of the input parameters. Furthermore, the DAML-S service description is used to dynamically build up a filtering panel where constraints on various properties of the service may be entered.

## 6. Conclusions and future work

In this paper we have presented an infrastructure that enables a large number of heterogeneous Registries to provide publication and discovery of an unlimited number of services. The services are categorized and maintained in domain-based Registries. Service categorization solves the scalability problem, helps in narrowing the search context and improves performance. The use of syntactic, semantic as well as QoS information about a service enables result ranking and service selection. The introduction of Gateways as an entry point to the system abstracts the interface of different discovery mechanisms into a single unified view. This entails the following advantages:

- PYRAMID-S integrates all types of WS Registries achieving thereby better interoperability without affecting their specifications and autonomy.
- PYRAMID-S is an open platform as it supports a variety of discovery mechanisms, and so it is useful irrespective of which protocols are finally adopted.
- By using the SQ and SA templates and the interfaces provided by the Gateways, the access to the different Registries and the homogenization across different representations is transparent to the user (both developer and end-user).

A prototype implementation of PYRAMID-S is under way. After the completion and the performance analysis of the current implementation, we are considering investigating the following enhancements:

- Security measures during Registry addition in the DCO.
- Use of Content Distribution Networks (CDN) [21] to improve load balancing and performance in the Gateways layer of PYRAMID-S.

We would like to point out that the viability of our approach depends on two important issues: efficient ontology design and management as well as appropriate regulation for the involved economic and social aspects. More precisely, designing and maintaining ontologies is a time consuming activity as it requires agreement and consensus among domain experts. Thus, in the scope of

PYRAMID-S several issues arise such as who is going to define the ontologies, who will check the validity of a new domain or alteration of a domain in the DCO.

Regarding the social and economic issues associated with PYRAMID-S there are several questions that need to be answered such as: Who operates the Routers and Gateways? What benefits do they receive as a return on their costs in establishing and operating them? Surely, rewarding financially the operators of the Gateways and the Routers will result in better and more reliable services. Several solutions may be envisioned such as operators funded by their corresponding governments or by the various registries, however this is beyond the scope of this paper.

## Acknowledgement

This work has been partially funded by the Special Account for Research (ELKE) of the University of Athens, under contract number 70/4/5829.

## References

- [1] DAML-S 0.9, <http://www.daml.org/services/DAML-S/0.9/>
- [2] DAML Services Coalition, DAML-S: Web Service Description for the Semantic Web, In the Proceedings of The First International Semantic Web Conference (ISWC), Sardinia (Italy), June, 2002
- [3] DAML Services Coalition, DAML-S: Semantic Markup for Web Services, In Proceedings of the International Semantic Web Working Symposium (SWWS), July 30-August 1, 2001
- [4] ebXML Registry, [http://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=regrep](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=regrep)
- [5] Sivashanmugam, K., Verma, K., Sheth, A., Miller, J., Adding Semantics to Web Services Standards, In Proceedings of 2003 International Conference on Web Services (ICWS'03)
- [6] Verma, K., Sivashanmugam, K., Sheth, A., Patil, A., Oundhakar, S. and Miller, J., METEOR-S WSDI: A Scalable Infrastructure of Registries for Semantic Publication and Discovery of Web Services, *Journal of Information Technology and Management (to appear)*
- [7] Paolucci, M. and Kawamura, T. and Payne, T.R. and Sycara, K., Importing the Semantic Web in UDDI, In Proceedings of Web Services, E-Business and Semantic Web Workshop, CaiSE 2002., pages 225-236, Toronto, Canada
- [8] Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K. Semantic Matching of Web Services Capabilities, In: Proceedings of the 1st International Semantic Web Conference (ISWC), 2002
- [9] Cardoso, J. and Sheth, A., Semantic e-workflow composition, Technical Report UGA-CS-TR-02-004, LSDIS Lab, Computer Science Department, University of Georgia
- [10] Cardoso J., Miller J., Sheth A. and Arnold J., Modeling Quality of Service for Workflows and Web Service Processes. Technical Report UGA-CS-TR-02-002, LSDIS Lab, Computer Science Department, University of Georgia
- [11] Sheth A., Cardoso J., Miller J., Kochut K. and Kang M., QoS for Service-Oriented Middleware. The 6th World

Multiconference on Systemics, Cybernetics and Informatics, Proceedings Vol. 8, Orlando, FL (Invited Session on Web Services & Grid Computing ), July 14-18, 2002, pp. 528-534

[12] Paolucci, M., Sycara, K., Nishimura, K., and Srinivasan, N., Toward a Semantic Web e-commerce, To appear in Proceedings of Business Information Systems (BIS2003), Colorado Springs, USA

[13] McIlraith, S., Son, T.C., and Zeng, H., Semantic Web Services, IEEE Intelligent Systems, Special Issue on the Semantic Web, Volume 16, No. 2, pp. 46-53, March/April, 2001

[14] Sirin, E., Hendler, H., Parsia, H., Semi-automatic Composition of Web Services using Semantic Descriptions, In the Proceedings of Web Services: Modeling, Architecture and Infrastructure Workshop in conjunction with ICEIS2003, 2002

[15] Shuping, R., A Model for Web Services Discovery with QoS, SIGecom Exchanges, Vol. 4.1, Spring 2003

[16] Web Services Architecture Requirements.

<http://www.w3.org/TR/2002/WD-wsa-reqs-20021114>

[17] Web Services Description Language.

<http://www.w3.org/TR/2002/WD-wsdl12-20020709/>

[18] Universal Description, Discovery and Integration.

<http://www.uddi.org>.

[19] Web Ontology Language (OWL).

<http://www.w3.org/TR/2002/WD-owl-user-interfacede-20021104/>

[20] ShaikhAli, A., Rana, O. F., Al-Ali, R., Walker, D. W., UDDIe: An Extended Registry for Web Services, In Proceedings of 2003 Symposium on Applications and the Internet Workshops (SAINT'03 Workshops) January 27 - 31, 2003, Orlando, Florida

[21] Rabinovich, M. and Spatscheck, O., Web Caching and Replication , Addison-Wesley 2001