Discovery of Web Services in a Federated Registry Environment

Kaarthik Sivashanmugam, Kunal Verma, Amit Sheth
Large Scale Distributed Information Systems (LSDIS) Lab
Department of Computer Science, University of Georgia
Athens, GA 30602
{verma, amit}@cs.uga.edu

Abstract

The potential of a large scale growth of private and semi-private registries is creating the need for an infrastructure which can support discovery and publication over a group of autonomous registries. Recent versions of UDDI have made changes to accommodate interactions between distributed registries. In this paper, we discuss METEOR-S Web Service Discovery Infrastructure, which provides an ontology based infrastructure to provide access to registries divided based on business domains and grouped into federations. We also discuss how Web service discovery is carried out within a federation.

Keywords: Web services Discovery, peer to peer, Decentralized UDDI

1. Introduction

There has been a significant change in focus of the vision of UDDI [UDDI, 2002] since its inception. This was evident in the release of version 3 [UDDI, 2003], which has several new features to augment the centralized paradigm of UBR to facilitate interaction between the UBR and private and semi-private registries. The current search facilities offered by the latest version of UDDI do not offer any special features for finding Web service registries. As a result, it is assumed that Web service clients have prior knowledge of the location of the registries. In this paper, we present our implementation of a peer to peer network of private semi-private and public UDDI registries, which allows transparent access to other registries based on registry federations or domains. We use an ontology based approach to classify registries and locate them based on the users' requirements.

Let us consider the following scenario which illustrates the benefits of private registries and having the ability to interact with other private registries. We can imagine a manufacturer that maintains a private registry to maintain details about its suppliers and other partners. Now consider a case when its suppliers are unable to meet its demands either due to adverse circumstances or large orders, and the manufacturer has to locate other suppliers. Due to trust issues, the manufacturer may not want to search the UBR and find just any supplier. It may however want to request its partners or competitors for references about the trusted suppliers or it may want to contact a marketplace to find similar services. Assuming that partners maintain similar private registries, this process can be automated by forming registries federations, where the registry owners give only members of the federation access to their registries. Forming a federation of registries will allow businesses to share their data while maintaining their privacy.

In this paper, we leverage the METEOR-S Web Service Discovery Infrastructure (MWSDI) [Verma et al., 2004] for providing transparent access to private and public Web service registries. The focus of this paper is the creation of registry federations. We present a discussion of registry federations and characterize them in the dimensions of distribution, autonomy and heterogeneity. In order provide efficient access to the registries we store semantic metadata Web service registry community in the Extended Registries Ontology (XTRO). We also discuss the different kinds of querying possible using our infrastructure. This work was done as part of the METEOR-S project, which aims to create an infrastructure for complete lifecycle management of Semantic Web processes.

We briefly describe the need for decentralization in section 2. The technical details of MWSDI are described in section 3. The classes and relationships of the XTRO are shown in section 4. In section 5, we discuss and analyze Registry Federations. In section 6, we have discussed implementation details in distributed registries with the help of tModels. Service publication and discovery have been discussed in sections 7 and 8. Section 9 discusses related work. Conclusions and future work are mentioned in section 10.

2. Need for Decentralization

One of the main reasons that the UDDI specifications decided to acknowledge replication model for data in registries instead of distribution is to enable inspection of data in UDDI along multiple perspectives. There can be potentially several facets to distribute data in UDDI, such as those related to geographical location, nature of registered services, business functionality, technical specifications and so on. The data partitioning on the other dimension could be hierarchical or non-hierarchical. With such distribution architecture, it should be possible not

only to locate appropriate registry (by processing all kinds of data distribution facets and data partitioning criteria) but also to aggregate search results from different candidate registries. Replication was chosen in UDDI because creating a scalable model for distribution of data is inherently difficult. However, in version 3 of the UDDI specifications, the need for data partitioning and affiliation among registries have been acknowledged. In MWSDI, we advocate data distribution (as opposed to data replication) and support any kind of data distribution among multiple registries[1]. The data partitioning criteria is stored in the *Extended Registries Ontology* (XTRO) in MWSDI.

3. MWSDI

In this section, we briefly describe the conceptual foundations and implementation of MWSDI which provides the infrastructure for the work presented in this paper. Detailed descriptions of the architecture, implementation and protocols can be found in [Verma et al., 2004]. The aim of MWSDI is to provide clients with an efficient publication and discovery mechanism in a multi registry environment. We use semantic metadata stored in the XTRO to identify appropriate registries and direct the queries to them. Each time, a new registry is added to MWSDI, the XTRO is updated with the relevant details of the new registry. We show the interaction of MWSDI, registries and clients in Figure 1.

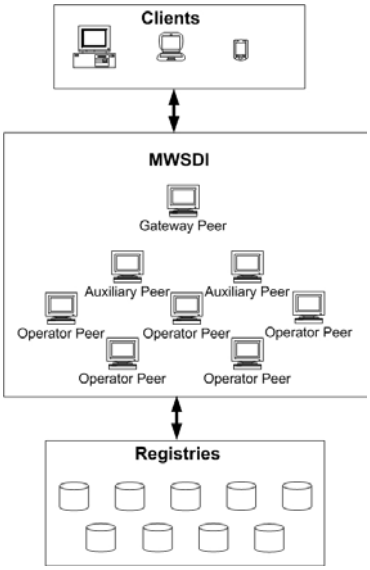

Figure 1: Interaction of MWSDI with clients and registries

[1] Our prototype implementation is tested with non-hierarchical distribution of data. Data replication is also supported using the "replicateOf" relationship in XTRO.

We have implemented MWSDI as a peer to peer network. Based on the different kinds of functionality, we have implemented different peer types. They are the following.

Gateway Peer: Gateway Peer acts as an entry point for registries to join MWSDI. It is responsible for updating the XTRO when new registries join the network. Gateway Peer is the only peer that can update the XTRO or initiate new peers. It is also responsible for propagating any updates in the XTRO to all the other peers.

Operator Peer: The role of the Operator Peer is to operate a UDDI registry and to provide Operator Services for its registry. Operator Services are the value added services like semantic discovery and publication of Web services, provided by the registry operators. The Operator Peer also acts as a provider for the XTRO to all other peers who need it.

Auxiliary Peer: Auxiliary Peers act as providers of the XTRO to make it highly available which is critical to the performance of the infrastructure. In event of failure of the Gateway Peer, one of the auxiliary peers starts to act as the Gateway peer.

Client Peer: The Client Peers are transient members of the peer-to-peer network, as they are instantiated only to allow users to utilize the capabilities of the MWSDI.

MWSDI has been implemented on a cluster of SUN workstations as peer to peer network using the JXTA [JXTA] framework. Any peer can be a JXTA peer if it implements one or more JXTA protocols. While there are a number of such protocols, we have used the Peer Discovery Protocol and the Pipe Binding Protocol. In addition to the protocols available in JXTA framework we have implemented two MWSDI specific protocols. They are:

Operator Peer Initiation Protocol: It defines the protocol involved in adding a new registry to the MWSDI system. It involves creating a new registry instance in XTRO. In some cases it may involve creating new federation or a domain.

Client Peer Interaction Protocol: It defines the protocol for accessing registries for publication and discovery. Details of these protocols are available in [Verma et al., 2004].

4. Extended Registries Ontology (XTRO)

In our initial, naïve implementation registries could only be categorized based on business domains. We did not create constructs for creating registry federations. The *Registries Ontology* in our initial implementation is extended (and called as *Extended Registries Ontology* or XTRO) in this work that supports complex classification as well as Registry Federations. Extended Registries ontology (XTRO), represented in OWL, is a comprehensive ontology containing details of Domains,

Registries, Ontologies and Registry Federation and network of relationships among them. All the classes and few important object properties in XTRO are shown in Figure 2.

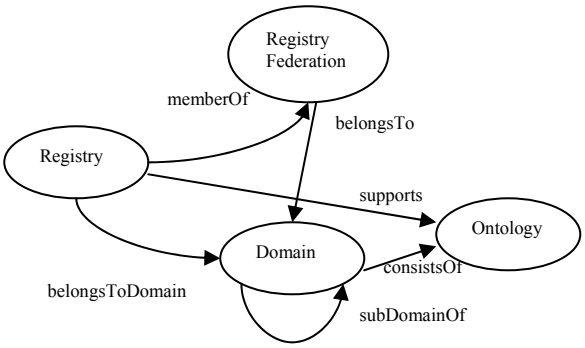

Figure 2: Classes and their Relationships in XTRO

Let us briefly examine the classes and their relationships shown in Figure 2. Each registry in the network is an instance of the class *Registry*. Different business domains are represented as instances of the class *Domain*. Ontologies available in MWSDI are instances of class *Ontology* and registry federations in MWSDI are instances of the class *RegistryFederation*. Instances of *Registry* and *RegistryFederation* can belong to one or more domains. A *Domain* instance has a number of *Ontology* instances to describe it. A *Registry* instance can support one or more domain ontologies and a *Domain* is usually a specialization of a more generic *Domain*. Each class has a number of attributes which give us more information about their instances. We have not included them in this paper for brevity. This simple set of classes and relationships provides a simplified but exhaustive view of the Web service registry community which can be used for efficient and accurate selection of registries. The XTRO allows us to maintain information and provide answers for the following type of queries to identify one or a group of registries:

- What is the access URL, available data model or type of the registry *R*?
- Does the registry *R* support the ontology *O*?
- Which are the registries available under the business domain *B*?
- Is the registry *X* a member of the registry federation *Y*?
- Which registries pertain to the domains that support the ontologies *O1* and *O2*?
- Get all the registry federations that belong to the domain *D*?
- Find all the registries that are categorized under the node *N* in the taxonomy (or ontology) *C*?
- What are the available relationships between the registry *R* and registry *S*?

With the use of such queries and their combination thereof to construct more complex queries, multi-perspective and intelligent querying can be carried out to locate registries for publishing or discovering Web services. We have shown a visualization of the XTRO in Figure 3.

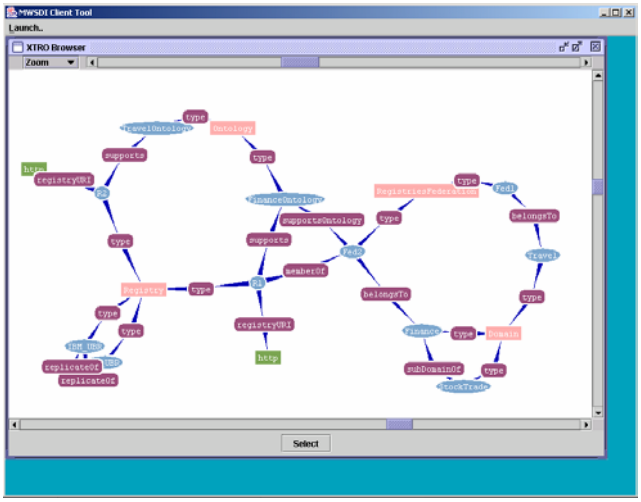

Figure 3: Visualization of XTRO

5. Registries Federation

We define a Registry Federation to be a collection of autonomous but cooperating Web service registries. The goal of a federation can be forming a registry community serving either a business domain or forming a market place of registries with similar but competing services. A federation can also aim to form an association of registries with interdependent services bound by trust and common network identity for confident collaboration. It can be a collection of private registries, public registries or even electronic marketplaces. The members of the registry federation can be heterogeneous and can have different data models and access APIs. A registry can participate in more than one federation. In the following sections, we explore the benefits of federations and characterize them with respect to the dimensions of distribution, heterogeneity and autonomy. Some of the the following issues have not been discussed by us in this paper.

- Comparison of the different federation models.
- Proposition of an architecture to establish interoperation between data models of heterogeneous Web service registries like UDDI and ebXML.
- Analysis or implementation of typical features needed in a federation (like identity management, trust/contract issues and business aspects).

5.1. Need for Creating Registry Federations

Federation of registries can provide several advantages over individual registries. The benefits of a typical federated system include, but are not limited to the following:

- Managing distributed information infrastructure (sharing data, establishing inter-system data references/dependencies) without integrating the information into a single system.
- Accessing and integrating information from disparate systems without having to hop around these information repositories and manually integrate the information from each. It also helps in achieving data model transparency.
- Providing a scalable approach in accommodating new information sources to the existing applications independent of the technology, data structure, API or version of the information source.
- Providing value added services (like common authentication/authorization methods, QoS provisioning, interfaces to specify query and information integration rules).

5.2. Characteristics of Registries Federation

The dimensions of distribution, heterogeneity and autonomy used for characterizing different types of federations of database systems where discussed in [Sheth and Larson 1990]. The same dimensions can be used to discuss characteristics of federation of registries. This work is not an attempt to discuss the intricate details of these characteristics. For the sake of clarity and completion, we have provided a brief description.

5.2.1. Data Distribution

Data may be distributed across several UDDI registries either in hierarchical or non-hierarchical models. This means that the entries in a UDDI need not be duplicated in all other registries. Current UDDI standard conforms to replication model, where all registries are equal and each registry is an exact and complete replica of each other. Hence executing a query in a registry would return the same result as that of executing the same query in another registry. Several researchers [Thaden et al., 2003, Schmidt and Parashkar, 2003] have argued that this kind of replication is not scalable. These research works also argue that distributing data among multiple registries based on vertical or horizontal partitions would provide increased availability and reliability.

5.2.2. Structural and Semantic Heterogeneity

The registries in a federation may display two types of heterogeneity, namely structural and semantic heterogeneities. Structural heterogeneity includes difference in data model. For example, an ebXML or a UDDI registry can be used as a Web service registry each with different data model. Within the same registry specification, registries could expose difference in terms of the data model and/or API versions they support. Semantic heterogeneity includes semantic differences in the elements of the data model**.** A typical federation allows the coexistence of the registries in a mutually beneficial and harmonious fashion in spite of the exhibited heterogeneities.

5.2.3. Autonomy

The registries in the federation will typically be autonomous. As a result, a registry operator may have a separate and independent control over his registry. Though autonomy could be described for various criteria there are two basic types of autonomy based on which registries could be defined. There are:

- Design autonomy: This includes selection of data model, API selection and different types of access, different algorithms for semantic publication and discovery.
- Execution autonomy: A registry may be able to support publication and discovery of Web services independent of other registries in the federation. Hence the publication and discovery mechanisms in a registry may be unaffected by whether or not a registry is a part of a federation.

We characterize the federations supported by MWSDI in the following manner. Our implementation allows data distribution with the help of XTRO. We support design autonomy as different registries can have different algorithms for semantic publication and discovery. We also support execution autonomy as registries in MWSDI can be accessed in a standalone manner without using any MWSDI components. We support limited form of semantic heterogeneity in the federation with the help of tModel directories discussed in the next section. However, we do not support structural heterogeneity, as we only support UDDI registries.

6. TModels, UDDI Registry and Federation of UDDI Registries

TModels are reusable metadata constructs in UDDI data structure that are used to characterize and categorize

businesses and their services. TModels provide the ability to describe compliance with a specification, a concept or a shared understanding. One of the main uses of a tModel is to define abstract namespace references. This means that a tModel can act as a reference that represents a relationship between keyed name-value pair and a namespace where the name-value pair has a meaning. With this characteristic, tModels are useful to annotate or attach categorization and identification information to the UDDI data. MWSDI utilizes this feature of tModels to add semantic annotations to the UDDI entries. Once tModels are registered to represent an idea or shared meaning and the annotations are added to the UDDI data using the registered tModels, the reference (tModel keys) to the tModels can be used to discover information in UDDI that are associated with the tModels. However, to perform matching based on keyed references, the tModel keys have to be specified. As a result, a query directed to UDDI registry cannot be used to query another registry. This is because, even if the same idea or shared specification is registered in two different registries as tModels, the tModel keys could (and in most cases it will) be different. Hence, to run a query across multiple UDDI registries, some kind of query parameter translation has to be done to ensure uniform semantic interpretation of the query across all the registries. Query parameter translation in MWSDI involves translating the tModel keys used in the query. If the query sent to registry R1 references to a tModel TM1 (with the key TK1) with a value V1 (in the name-value pair), representing a concept C1 in a taxonomy, ontology or some other shared specification, then to run the same query in another registry R2 without altering the query semantics, we have to replace the key TK1 in the query with a corresponding key TK2 which is an identifier (key) for the tModel TM2 (in R2) representing the same concept C1. The value V1 should not be changed to retain the semantics of the original query. Our implementation of federation of registries is based on the idea of maintaining mapping between tModel keys of tModels with same semantics across multiple registries and to use it for federated search.

In MWSDI, each federation will have a tModel directory. The tModel directory is a simple special purpose UDDI registry that registers only tModel related data. For every unique tModel across the registries in the federation, there will be a representative tModel registered in the tModel directory. Each of the representative tModel stores mapping between registries and the corresponding tModel keys. For example, if there are three registries in the federation namely R1, R2 and R3 and if there is a tModel representing an ontology is published across these registries with three different keys TK1, TK2 and TK3 respectively, then the tModel directory will have a representative namespace tModel that stores the mapping R1-TK1, R2-TK2 and R3-TK3. If there is a query that is sent to the registry R2 referring to the tModel key TK2, then to run the same query in the registry R1, the key TK2 in the query is replaced with TK1 and to run the query in the registry R3, TK2 is replaced with TK3. Figure 4 shows the structure of a sample representative tModel. It is categorized as a namespace tModel. It also stores generic name-value pairs to store mappings between registries and the keys of the tModels (in these registries) that are linked by the representative tModel.

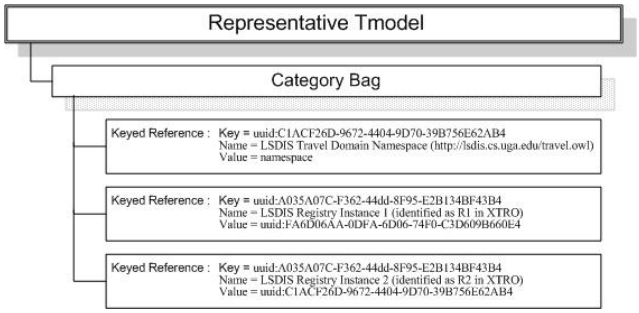

Figure 4: Representation of TModel Structure

6.1. Implementation of TModel Directory

Whenever a tModel is registered in the tModel directory as a representative of a tModel appearing in different registries with different keys, it will be categorized as a namespace tModel using the taxonomy value "namespace" in relation with one of the built-in core UDDI tModels which has the name "uddi-org:types", tModel key "uuid:C1ACF26D-9672-4404-9D70-39B756E62AB4". A keyed reference is constructed with this tModel key and the name-value pair is used for this categorization. The purpose of this categorization is to have the namespace functionality and to find all the representative tModels in the tModel directory that belong to a particular namespace. The representative tModel will also be categorized using another core built-in tModel with the name "uddi-org:general_keywords", tModel key "uuid:A035A07C-F362-44dd-8F95-E2B134BF43B4" that is used to store generic name value pairs. The name value pair represents the registry name and the key of the tModel in the registry that is associated with the representative tModel

6.2. Administration of the Federation

Federation of Registries can take place in two different ways. One way is to form the federation first, allowing only empty (non-populated or un-used) registries to join the federation wherein data in the registries are populated later and the other way is to form the federation that allows both used and un-used registries to join the federation. Version 3 specification of UDDI supports the former to establish affiliation between multiple registries with

appropriate policies to allow controlled copying of data structures among them. These registries share a common namespace for entity keys for the purpose of entity promotion and hence if a tModel has to be published in more than one registry it can be published with the same tModel key in all the affiliated registries. This kind of affiliation makes it practically difficult for a registry to be a part of more than one independent affiliation because namespace consistency for the keys are difficult to enforce across independent affiliations. In MWSDI, we support both ways of establishing registries federation. In addition our implementation can support building federation using any version of UDDI registry as the implementation uses core UDDI data structure (tModels mainly) and nothing outside of it. Also, it empowers a registry to take part in more than one federation as the membership of a registry in a federation does not affect its other operations including key assignment for its data. The unique feature about our federation is that a registry can arbitrarily join or leave the federation without affecting other registries or without affecting the existing applications that are dependent on keys of the UDDI data.

In MWSDI, the registries join federation one by one. When the first registry (say R1) which probably will be the initiator of the federation joins the federation, all the tModels in the registry are treated unique (i.e. unique overviewURLs are assumed). For every tModel in the registry that joins the federation, a representative tModel is registered in the tModel directory of the federation. Each of the representative tModel is categorized as namespace tModel. As the overview URL of each tModel (in R1) is unique, the namespace keyed reference in the category bag of the corresponding representative tModel can be used to hold this URL in the name-value pair. Each of the representative tModel is also categorized using a keyed reference (in conjunction with "uddi-org:general_keywords" taxonomy) representing the mapping between R1 and the key of the corresponding tModel in this registry. Hence a tModel (in R1) identified by TK1 will have a representative in the tModel directory that is categorized using the mapping R1-TK1. It should be noted that the structure of the tModels are not copied from the federation member to the tModel directory. The mapping in the directory acts as a pointer to the tModel in R1. When another registry (say R2) joins the federation, overview URL of each tModel is compared against the value (in name-value pair) of all the namespace keyed references in the directory. If, for a tModel (in R2) with key TK2, there is any existing namespace keyed reference for a representative tModel (in tModel directory), then a keyed reference representing the mapping between R2 and TK2 is added to the existing categorization of the representative tModel., Consider a representative tModel that already has the mapping between the registry R1 and the tModel with key TK1. After the federation figures out that the equivalent tModel in R2 has the key TK2, the system will add another keyed reference to represent the mapping between the registry R2 and the tModel with key TK2. In this way a tModel registered across multiple registries are linked using one representative tModel that is characterized using the name-value pair Rx-TKx where Rx represents a registry that hosts the tModel with a key TKx. After the federation is formed, whenever a tModel is published in one of the member registries, a similar step is undertaken to ensure updated mapping details between registries and tModel keys. Deleting a tModel in a registry removes a corresponding keyed reference in the category bag of the representative tModel in the directory.

7. Service Publication in a MWSDI-2G

Service publication in MWSDI involves the following steps.

Creating Service Advertisements: Annotation of WSDL files with the help of ontologies has been discussed in [Patil et al., 2004; Sivashanmugam et al., 2003]. Users can publish annotated or standard WSDL files using the client peers.

Registry Selection: In case of standard WSDL files, the user is required to manually specify the requirements for registry selection using the GUI tool shown in Figure 6. For semantic publication of Web services, the users can specify either the federation names or business domains. We have added the functionality in MWSDI to automatically determine the registries on the basis of this information and ontologies used for annotation. A registry selection query can be expressed as R, where

R = <*f*, *d*, *o*, *r*> and *f* corresponds to the names of the registry federations, *d* corresponds to the business domains, *o* corresponds to set of ontologies, and *r* corresponds to a set of registry relationships. Let us assume that a particular query has the following values

R = < {F1,F2}, {D2, D3}, {O1, O2, O3, O4}, {} >

The results of such a query are shown in Figure 5. The registries are shown categorized in Federations F1, F2 and F3 as well as domains D1, D2 and D3. The shaded registries are results of the query. Some registries in the appropriate domains and federations are not selected as they do not support the ontologies in the query. Even though registry relationships have not been shown in the sample query, relationships of registries like *belongsTo* and *partnerOf* can be used to select more registries. We use the inference capabilities provided by SNOBASE [Lee et al., 2003] for the handling the queries.

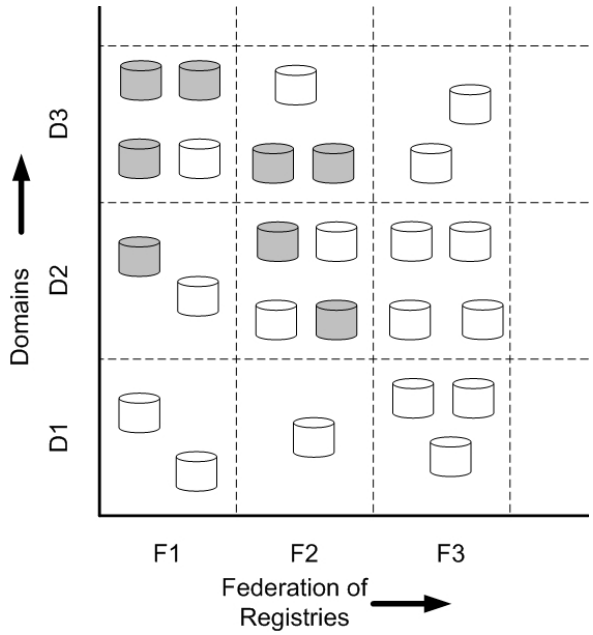

Figure 5: Selection of Registries

The GUI tool for constructing the query R is shown in Figure 6. The initial set of domains selected from browsing XTRO is shown in the top left panel. We provide the users with the option of filtering the domains before adding them to the query. The final selection of domains is shown in the top right panel. The available relationships between domains and registries and shown in the bottom left panel. Other dimensions of the query, namely ontologies, federations and inter registry relationships can be specified using the tabs shown in Figure 6.

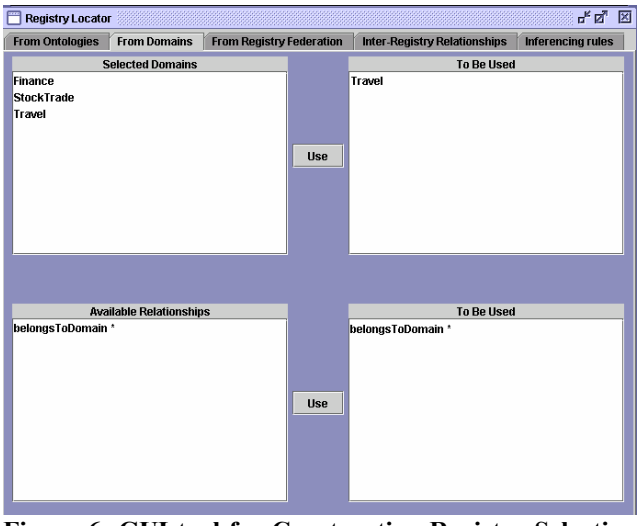

Figure 6: GUI tool for Constructing Registry Selection Query

Service Publication: The services are the published by sending the advertisements to the operator peers of the selected registries.

8. Service discovery in MWSDI-2G

MWSDI supports both semantic and syntactic discovery of services. Service discovery in MWSDI-2G involves following steps:

Creating Search Templates: Users can either use the search mechanism provided by UDDI or create semantic templates. We have discussed semantic templates in [Sivashanmugam et al., 2003;Sivashanmugam et al., 2003a]

Registry Selection: Registry selection is the same as discussed in the previous section on publication.

Query Execution: The service templates are then sent to the operator peers of the selected registries. We have provided an option of creating a "federated query". In case of the federated query, the TModel directory is used to translate and propagate the query to other registries in the federation.

Result aggregation: The operator peers return all the results to the client peer.

9. Related Work

The approaches to Web services discovery can be classified as centralized and decentralized. UDDI falls under fully centralized approach that supports replication. Having realized that replicating the UDDI data is not a scalable approach several decentralized approaches have been proposed. [Thaden et al., 2003] argues that the trend in integrating UDDI features into general purpose enterprise registries results in rapid increase of private registries and limits the use of public registries. It also states that, from discovery perspective, it is impractical to replicate these private registries in the public counterparts. They propose to deal with this problem by creating a virtual global registry by connecting all private registries in a P2P network. They also support semantics based service discovery using DAML-S service descriptions and matchmaking. This work is similar to our work but they do not consider registry federations. [Schlosser et al., 2002] details the use of a global ontology to determine the organization of peers in their P2P topology thus enabling concept based search. It is about ontology based clustering of peers based on their capabilities. P2P based web service discovery is also discussed in [Schmidt and Parashkar, 2003; Paolucci et al., 2003; Maedche and Staab, 2003]. None of the aforementioned works considers relationships between registries and registry federations and routing queries on the basis of them. Our work is different as due we use XTRO to capture relationships among registries as

well caterorize the registries on the basis of business domains.

[Zhou et al., 2003] presents an federated architecture that supports QoS based discovery of services. It has a notion of UX ("UDDI Extension") server that performs federated discovery on behalf of a user request and aggregates results before sending them back to the requestor. The paper discusses different ways of maintaining links between the servers and how query is propagated. It also envisions linking UX servers across different domains. It points out that improvements could be made using semantic descriptions and matchmaking. In our approach, we have used JXTA and abstracted the network level issues. The emphasis is rather on utilizing UDDI data structure to store semantic description of a service for better service matchmaking, to establish a federation for carrying out discovery process in multiple registries and in exploiting an ontology for improving the registry selection mechanism for Web service publication and discovery.

Looking from a business perspective our work shares that the perspective of [Christoffel, 2001]. It emphasizes the idea of increasing the potential of individual traders by cooperation with other traders. It also argues that cooperation between traders can be useful when a trader cannot provide sufficient service to a customer. It also discusses the use of centralized and non-centralized federations. Our work applies this idea to Web service registries and argues the use of cooperation of registries where in registries can work together as a federation to get useful results during Web service discovery. Due to existence of tModel directory in federations, our federation can be termed as centralized federation. Federated Web service discovery is discussed in [Chandana et al., 2003]. However, it is limited to keyword based search on a predefined set of UDDI registries. Business Explorer for Web Services (BE4WS) [BE4WS, 2001] is an XML-based UDDI exploring engine to perform complex searches using a single query request on a single or multiple UDDI Registries. It does not however support tModel translation while performing a query across multiple registries as discussed in our work.

10. Conclusion and Future work

In this paper we discussed how the metadata of a registry network stored in an ontology (XTRO) can be used in registry selection to perform Web service publication or discovery. [UDDIDS, 2002] says that data in UDDI is not sufficient to provide searches to find partners with products based on price range or availability, or to find high quality partners. With XTRO as a common resource in the P2P based architecture and the support for semantics based publication and discovery of Web services in MWSDI, it would be possible for applications, businesses, partners and suppliers to analyze information about the registries and their characteristics to effectively carry out the process of service discovery.

In the future we intend to extend this work to incorporate the following:

- Establishing contracts, trust and security policies among members of the federation
- Mediator to enhance interaction between federations
- Dynamic/Runtime association between registries

References

[BE4WS, 2001] Business Explorer for Web Services, http://www.alphaworks.ibm.com/tech/be4ws

[Chandana et al., 2003] C. Subasinghe, D. Cooray, N. Sadeep, L. Kumara, Wonder Room for Easy Web Services Invocation, Department of Computer Science and Engineering University of Moratuwa, December 2003.

[Christoffel, 2001] M. Christoffel. Cooperations and Federations of Traders in an Information Market. In Proceedings of the AISB Symposium Intelligent Agents in Electronic Commerce, York, 2001.

[JXTA] http://www.jxta.org

[Lee et al., 2003] J. Lee , R. Goodwin, R. Akkiraju, P. Doshi, Y. Ye 2003 SNoBASE: A Semantic Network-based Ontology Ontology Management. http://alphaWorks.ibm.com/tech/snobase.

[Maedche and Staab, 2003] A. Maedche, S. Staab. Services on the Move - Towards P2P-Enabled Semantic Web Services. In: Proceedings of the 10th International Conference on Information Technology and Travel & Tourism, ENTER 2003, Helsinki, Finland, 29th-31st January 2003.

[Paolucci et al., 2003] M. Paolucci, K. Sycara, T. Nishimura, and N. Srinivasan, "Using DAML-S for P2P Discovery," in Proceedings of the First International Conference on Web Services (ICWS'03), Las Vegas, Nevada, USA, June 2003, pp 203- 207 .

[Schlosser et al., 2002] M. Schlosser, M. Sintek, S. Decker and W. Nejdl. A Scalable and Ontology-Based P2P Infrastructure for Semantic Web Services. Second International Conference on Peer-to-Peer Computing (P2P'02) September 05-07, Linkoping, Sweden 2002.

[Schmidt and Parashkar, 2003] A Peer-to-Peer Approach to Web Service Discovery, C. Schmidt and M. Parashar,

World Wide Web, Internet and Web Information Systems, Kluwer Academic Publishers, August 2003.

[Sivashanmugam et al., 2003] K. Sivashanmugam, K. Verma, A. Sheth, J. Miller, Adding Semantics to Web Services Standards, Proceedings of the 1st International Conference on Web Services (ICWS'03), Las Vegas, Nevada (June 2003) pp. 395-401.

[Sivashanmugam et al., 2003a] K. Sivashanmugam, J. Miller , A. Sheth, K. Verma 2003. *Technical Report 03-008*, LSDIS Lab, Computer Science Dept., University of Georgia.

[Thaden et al, 2003] U. Thaden, W. Siberski, and W. Nejdl, A Semantic Web based Peer-to-Peer Service Registry Network, Technical Report, Learning Lab Lower Saxony,

[UDDI, 2002] Evolution of UDDI, White Paper available at http://www.uddi.org/pubs/the_evolution_of_uddi_20020719.pdf

[UDDIDS, 2002] UDDI Version 2.03 Data Structure Reference, UDDI Committee Specification, 19 July 2002 http://uddi.org/pubs/DataStructure-V2.03-Published-20020719.htm

[UDDI, 2003] UDDI Version 3.0.1, available at http://uddi.org/pubs/uddi-v3.0.1-20031014.htm

[Verma et al., 2004] K. Verma, K. Sivashanmugam, A. Sheth, A. Patil, S. Oundhakar and J. Miller, METEOR–S WSDI: A Scalable Infrastructure of Registries for Semantic Publication and Discovery of Web Services, Journal of Information Technology and Management (to appear, 2004)

[Zhou et al., 2003] Zhou Chen, Chia Liang-Tien, Bilhanan Silverajan and Lee Bu-Sung, UX- An Architecture Providing QoS-Aware and Federated Support for UDDI, In proceeding of the first International Conference on Web Services(ICWS03), 2003