CHAPTER 1
INTRODUCTION

1.1 OVERVIEW

A web service can be thought of as a web application which uses XML based standards for communicating with external systems for providing the necessary service to the user. The Web Service Architecture working group[1] defines a web service as a software system designed to support interoperable machine-to-machine interaction over a network and has an interface described in a machine-processable format (specifically WSDL) where other systems interact with the system in a manner prescribed by its description using SOAP messages. Web Services are encapsulated, loosely coupled, self-describing, self-advertising, uniquely addressable, standards based and platform independent contracted functions offered through standard protocols [2][4].

- “Encapsulated” means the implementation of function is never seen from the outside.
- “Loosely coupled” means changing the implementation of one function does not require change of the invoking function.
- “Self-describing” means the descriptions are made once the service is created, which provides the information about the operations provided by the service.
• “Self-describing” means the web services are made available in a public registry that can enable the clients to search.

• “Uniquely addressable” means every web service has a uniform resource identifier which can be located using internet protocols.

• “Standards based” refers to the technologies that enable web services which form a base for comparison.

• “Platform independent” means web services do not depend on any specific technological platform.

• “Contracted” means there are publicly available descriptions of the function’s behaviour, like how to bind to the function as well as its input and output parameters.

Hence the software system called Web Services has varied features and functions.

The following Table 1.1 describes some of the characteristics of web services and the technology that enables these functionalities to be achieved.

<table>
<thead>
<tr>
<th>Web Service Characteristics</th>
<th>Underlying Technology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-Describing</td>
<td>WSDL</td>
</tr>
<tr>
<td>Self-Advertising</td>
<td>UDDI</td>
</tr>
<tr>
<td>Uniquely Addressable</td>
<td>Internet Protocols</td>
</tr>
<tr>
<td>Standards Based</td>
<td>XML, SOAP, WSDL and UDDI</td>
</tr>
<tr>
<td>Platform Independent</td>
<td>XML and SOAP</td>
</tr>
</tbody>
</table>
1.2 WEB SERVICE COMPUTING MODEL

Web service computing model are different from traditional distributed computing models. Web service architectures provide a framework for creating and deploying loosely coupled applications. One of the consequences of loose coupling is that any entity that a web service may interact with may not exist at the point of time the web service is developed. New web services may be created dynamically just as new web pages are added to the web and web services should be able to discover and invoke such services without recompiling or changing any line of code.

The web service architecture allows businesses to expose business assets as services. Standardizing interactions amongst services has the added advantage that any enterprise can out-source parts of its operation that it does not have expertise in. In addition, since the vision of web services enables web services to dynamically find new web services that it can interact with, enterprises can find new providers for the service relatively quickly. A specific application of this dynamism is in the e-procurement arena. For example, the average sourcing/procurement cycle in enterprises is of the order of 3-4 months. Of this time, about 50% of the time is spent in identifying the appropriate suppliers, about 20% of the time in handling the RFQ (request for quotes) process, and an additional 10% of the time is spent in negotiating the appropriate deal. The ability to dynamically find suppliers can translate to
significant time savings, and therefore to lowering of costs. Essentially, the procurement and fulfillment business process are modeled as services, and a hub is the aggregation point for the services. In such architecture, finding a new supplier is the same as finding the fulfillment service of the supplier at the hub.

Web services have their own unique features. They are fundamentally different from traditional distributed models for the following reasons:

- **Web services are loosely coupled**: Changes to a web service should not require re-installation of software components by the users of the web service.

- **Web services require dynamic binding**: Typically, application designers bind software components to one another at development time. Web services, on the other hand, are likely to be implemented and provided by different service providers. In addition, it is a must to enable easy changes to the services being used, easy discovery of new services, new capabilities of existing services, and new binding or location information of services.

- **Web services communication is based on a Document Exchange Model**: Traditional component frameworks support a network-object model of interaction in which objects of strictly defined types are transferred between components using a request-response interaction pattern. Cross-organizational business interactions do not fit this
framework well for two reasons. The interfaces of services may need to be changed in ways that cannot be captured by simple extensions. This precludes the use of object inheritance to support the inter-operability in presence of change. Secondly, interactions can be long lived. Therefore, asynchronous exchange of XML documents is better suited for cross-organizational business transactions.

- **Web services in different enterprises are likely to use different semantics.** The interpretation of the data communicated among enterprises is different for each enterprise. For instance, the address field of a purchase order may have different significance for the parties. If a uniform object model is used, the semantics of data often tends to be similar or homogeneous contributing to tighter coupling.

- **Web services in different enterprises require a distributed model of security.** Security responsibilities are split amongst the enterprises. Each enterprise manages its end of the security infrastructure independently.

- **Web services from different enterprises may be built using heterogeneous technology stacks.** Each enterprise decides on the computing infrastructure independently taking into consideration many factors.

- **Web service interactions must be able to traverse corporate firewalls.** Traditional distributed systems are tuned for applications that are deployed within the enterprise; Web services may be deployed from
behind Firewalls. They may need to access other web services across enterprises.

Web services are a standardized way of establishing communication between two web-based applications by using XML, SOAP, WSDL, UDDI, and open standards over an Internet protocol backbone. Where XML is used to encode the data in the form of SOAP message, SOAP is used to exchange information over HTTP, WSDL is used to describe the capabilities of web services, and UDDI is used to provide the list of service provider details, as shown in Fig 1.1.

Figure 1.1 Web Service Components
In the real-time scenario, if a service consumer wants to use some sort of web services then he/she must know the service provider. If a service provider validates a service consumer, it will provide the WSDL file directly and then the service consumers create a XML message to request the required service in the form of a SOAP message and the service provider returns a service response.

On the other hand, if the service consumer is not aware of the service provider, he/she will visit UDDI and search for the required service. The UDDI returns the list of service providers offering that particular service. Then, by choosing one service provider, the service consumer again generates an XML message to request the required service in the form of a SOAP message, as specified in the WSDL file of the service provider, and the service provider returns a service response.

The elements of the computing model can be elaborated by understanding the operations and the components shown in the Figure 1.1 which can be termed as a detailed view of the web service computing model.

1.2.1 Service Provider

The service provider is the actual owner of the web service from the business point of view. The web service is created by the service provider and it is given a uniform resource identifier which enables the users to use the service. The service provider can also be described as a platform that hosts the web services [3].
1.2.2 Service Registry (UDDI)

The service registry is the central registry of web services where the service providers publish their service, which can be searched by the service requestors. This registry is often referred to as Universal Description Discovery and Integration (UDDI).

1.2.3 Service Requestor (Service Consumer)

The service requestor maybe the actual user of the web service or a broker that uses the web service to satisfy the request of its client. The service requestor can be viewed as a business that requires certain functions to be satisfied from the business point of view and can be viewed as an application that is looking for initiating an interaction with a service from the application point of view. The service requestor is usually a web browser using which we communicate with the Internet. The service requestor can also be a standalone application or even another web service that requires a service [3].

1.3 WEB SERVICE CONCEPTUAL STACK

The web service computing model describes the roles, service provider, service requestor and service registry with the operations, publish, find and bind. In order to perform these operations by the roles in an interoperable manner, a web service conceptual stack, which embraces standards at each level is essential. The web services conceptual stack is presented below in Figure 1.2.
The conceptual stack presents a collection of standards used by the web services computing model. At each layer of the stack, we can realize the standards and protocols that are responsible for performing the operations. The network forms the foundation for all applications in the modern computing world and web services are no exception.

The web service must be available over the network, so that it could be easily used by a remote client machine that requires some of the functionalities to execute. The network used in web services is often based on an HTTP protocol, but other kinds of protocols, such as FTP, SMTP, JMS or IIOP are also used.

![Web Services Conceptual Stack](image)

Figure 1.2 Web Services Conceptual Stack
An XML based messaging layer is present at the top of the network layer that facilitates the XML based communications between the web services and their clients. The Simple Object Access Protocol (SOAP) is an XML based messaging protocol that facilitates the various operations of the web services. SOAP is a simple, lightweight protocol for exchanging information in a decentralized and distributed manner and it can also combine with other protocols easily [4]. It makes web services to fully adapt XML, which is interoperable and platform independent. The service descriptions are in the form of WSDL documents, which enable the client to contact the web service in a manner prescribed by the specifications.

1.4 WEB SERVICE DISCOVERY MODEL

Web services provide access to software systems over the Internet using standard protocols. In the most basic scenario there is a Web Service Provider that publishes a service and a Web Service Consumer that uses this service. Web Service Discovery is the process of finding a suitable Web Service for a given task. Web Service Discovery is "the act of locating a machine-processable description of a web service that may have been previously unknown and that meets certain functional criteria". The goal is to find an appropriate web service [1]. A discovery service is a service that facilitates the process of performing the discovery of web services from the service registry based on the requirements of the service requestor. It is a logical role, and could be performed by the requester agent, the provider agent or some other
agent. The Figure 1.3 (adapted from [1]) describes the process of engaging a web service when a discovery service is used.

The service engagement process using a discovery service proceeds in roughly the following steps:

1. The requester and provider entities "become known to each other":
   a. The discovery service somehow obtains both the Web Service Description (WSD) and an associated Functional Description (FD) of the service.
   b. The requester entity supplies criteria to the discovery service to select a Web Service Description based on its associated Functional Description capabilities and potentially other characteristics.
   c. The discovery service returns one or more Web Service Descriptions (or references to them) that meet the specified criteria. If multiple service descriptions are returned, the requester entity selects one, perhaps using additional criteria.

2. The requester and provider entities agree on the semantics (Sem) of the desired interaction. The important point is that the parties must agree on the semantics, regardless of how that is achieved.
3. The service description and semantics are input to, or embodied in, both the requester agent and the provider agent, as appropriate.

4. The requester agent and provider agent exchange SOAP messages on behalf of their owners.

Figure 1.3 Web Service Discovery Model

To summarize the above discussion, web services possess various features which can be widely used in a distributed environment as they are based on XML
and its variants. The web services computing model is explored for understanding the components present in the entities. Some of the technologies that enable web services are also discussed in detail. The web services conceptual stack and Web Services Discovery Model are discussed in order to provide exposure to the various processes involved in web services.

1.4.1 Semantic Discovery

Semantic web services promise to add automation and dynamism to current web service technologies, considerably reducing the effort required to integrate applications, businesses and customers. One of the key tasks in the integration process is to locate services that can fulfill the application, business or customer needs. With current web service technologies, this is done mainly manually, which reduces the accuracy of the search and requires considerable effort. Semantic Web Services aim at providing formal descriptions of requests and web services that can be exploited to automate several tasks in the web services usage process, including dynamic discovery of services.

WSMO (Web Service Modelling Ontology) is a conceptual model for relevant aspects related to Semantic Web Services. It provides an ontology based framework, which supports the deployment and interoperability of Semantic Web Services. WSMO discovery provides a conceptual model for service discovery that exploits WSMO formal descriptions of goals and web services.
WSMO discovery [Keller et al., 2004a] defines a conceptual framework for locating services that totally or partially fulfil a requester goal. This conceptual framework distinguishes three major steps in discovery: goal discovery, web service discovery, and service discovery. Goal discovery is about abstracting a concrete user goal to a pre-defined, reusable, and formalized goal. Web service discovery deals with the matching of formalized goals and formalized web services and selecting web services that can potentially be used to get the desired service. In the last step, service discovery, the web services matched in the previous step are used to access the real services behind such web service interfaces, finally checking what services fulfil the requester goal.

1.4.2 Semantic Similarity Measures

Semantic similarity or semantic relatedness is a metric defined over a set of documents or terms, where the idea of distance between them is based on the likeness of their meaning or semantic content as opposed to similarity which can be estimated regarding their syntactical representation (e.g. their string format). These are mathematical tools used to estimate the strength of the semantic relationship between units of language, concepts or instances, through a numerical description obtained according to the comparison of information formally or implicitly supporting their meaning or describing their nature.
1.4.3 Cluster Based Discovery

Web services have emerged as one of the distributed computing technologies and sparked new interest in industrial and research communities. As Web services adopt open standard interfaces and protocols, they are likely to be used as basic software building blocks in service-oriented applications, which are expected to play an important role in a variety of application domains such as business application integration, business-to-business (B2B) and business information management. Meanwhile, inspired by the promise of applications presented by Web services, the research community has identified two major areas of interest: Web service discovery and Web service composition.

Web service discovery is normally defined as a matching process in which available service capabilities can satisfy a service requestor’s requirements. The capability of a Web service is often implicitly indicated through a service’s name, a method’s name and some descriptions included in the service. And this capability can be described as an abstract interface by using standard Web services Description Language (WSDL).

Although various approaches can be used to locate Web services on the web, this research is focused on the service discovery problem using the clustering method. The clustering methodology is a technology that transforms a complex problem into a series of simpler ones, which can be handled more easily. Specifically, this technology re-organizes a set of data into different groups based on
some standards of similarity. Clustering analysis has often been used in computer science, as in data mining, information retrieval, and pattern classification. A clustering analysis is applied to the filtering process by comparing services with related clusters.

1.4.4 Limitations in Web Services Discovery

The Web services set of standards is aimed at facilitating and improving the quality of component-based applications on the web. It consists of a set of related specifications, defining how reusable components should be specified (through the Web-Service Description Language – WSDL), how they should be advertised so that they can be discovered and reused (through the Universal Description, Discovery, and Integration API – UDDI), and how they should be invoked at run time (through the Simple Object Access Protocol API – SOAP).

The keyword-based discovery mechanism supported by UDDI and most existing service search engines, however, suffers from some key problems. Firstly, it is difficult for a user to obtain the desired services because the number of retrieved services with respect to the keywords may be huge. One of the possible solutions to this problem is to compress data for reducing the size of services returned to service requesters. However, conventional techniques such as Singular Value Decomposition (SVD) [3, 4] and Support Vector Machines (SVM) may not be suitable for dealing with a large document collection due to the high cost of computing and storage of SVD. Secondly, keywords are insufficient in expressing semantic concepts. This is
partially due to the fact that keywords are often described by natural language, which is much richer in terms of diversity. For example, syntactically different words may have similar semantics (synonyms) which results in low recall. In addition, semantically different concepts could possess identical representation (homonyms), leading to low precision. As a result, the retrieved services might be totally irrelevant to the need of their consumers.

The lack of semantics in description creates inefficiencies in exploiting the Web service discovery. Describing Web service with semantics provides the ability for automatic Web service discovery, invocation, composition and interoperation, and Web service execution monitoring. Current standards focus on syntactic description of Web services.

1.5 PERFORMANCE IMPROVEMENT IN WEB SERVICES DISCOVERY

Service discovery is the process of retrieving the service most similar to the query based on the description of functional and/or non-functional semantics. In the literature, some researchers deem service discovery systems as matchmakers; some cover the entire spectrum of tasks from service request to service invocation, which means the inclusion of the selection process. In general, any service discovery framework needs to define the matchmaker, which evaluates the similarity metric between two services. There are two kinds of results returned by the discovery system:
• **Exact discovery**: In this case, when a service satisfies exactly the user requirements.

• **Approximate discovery**: In the case when the discovery system returns a service (or set of services) which satisfies approximately the user requirements, considered a realistic case.

    The two types of discovery cited above encompass a large spectrum of possible match-making.

    The related work on service matching can be classified as two categories based on criteria:

    • **Category 1**: {logic-based matching, non-logic-based matching, hybrid matching}

    • **Category 2**: {interface level, process level, hybrid level}

    In the following section, the principles of each element of both categories are defined.

    • **Logic-based matching**: the matchmakers use the semantic relations and logic inference to measure the similarity between two services.

    The degree of logic-based matching can be determined either (a) exclusively within the considered logic theory by means of logic reasoning or (b) by a combination of logical inferences within the theory and algorithmic processing outside the theory [5].
• **Non-logic-based matching:** the matchmakers perform without any logic reasoning to determine the similarity between services. They use other techniques from the search area like graph isomorphism, information retrieval measurement etc.,

• **Hybrid matching:** the matchmakers use a combination of logic and non-logic mechanisms to perform the matching process.

• **Service profile matching level** (so-called black-box service matching): in this case, the matchmakers’ work is generally based on input/output matching.

• **Process matching level** (so-called glass-box service matching): in this case, the matchmakers use behaviour matching in terms of control and data flow.

• **Hybrid matching level:** the matchmakers use a combination of service profile matching level and process matching level mechanisms to perform the matching process.

The matching levels are so varied and meticulously defined that it is capable of returning a wide range of possible matching.
1.5.1 Performance Tuning in Semantic Similarity Measures

A discovery service is a service that facilitates the process of performing discovery of web services from the service registry based on the requirements of the service requestor. It is a logical role, and could be performed by the requester agent, the provider agent or some other agent. The service provider provides the services by registering them at the UDDI (Universal Description, Discovery and Integration) [1,24], which has to be discovered during service discovery. Though progressive and cutting-edge techniques have been proposed to address the service discovery challenges in conservative environment, still they are in their critical stage at the semantic discovery scenario, particularly when the quality attributes are included. The retrieved web services, at times, may not be desirable for the user and the objective of the user desirable service discovery fails.

In this context, Concept Lattice based clustering [10] technique is proposed as lattices provide the inherent relation between the web services that are in the UDDI as the web services are linked using hierarchical relation (operation in web service discovery). The representation of the web service using concept lattices is relatively a simple task as it can be modeled as graph data structure. The cost of time incurred for searching the cluster can be reduced using the proposed framework. The facility of concept lattice technique to include contextual knowledge can be of colossal use in discovering user desirable services.
1.5.2 Performance Tuning Web Services Clustering

Formal Concept Analysis is the process which mainly offers a formalization of mathematical concepts. The service which should be selected from the group of cluster is based on the usage of concept lattices to identify relationships between services or between service operations and to help understand types defined inside service interfaces. The present work builds concept lattices by applying formal concept analysis (FCA) to the documentation available for the considered set of services.

In this research, a framework for semantic discovery of services based on the properties clustered using concept lattice is developed based on multi-agent systems [2,3]. A novel similarity measure for assessing the semantic relevance during service discovery is proposed to bridge the semantic gap between the service request and the service provided. A two tier User Preferential Model (UPM) is proposed to support the service discovery with respect to the non-functional requests. The tier I of the UPM deals with the qualification of the QoS parameters, where the user is presented with the available quality parameters for defining them in the model. The tier II of the UPM quantifies the qualified QoS parameters, where the user will actually set the preference values. Concept lattices can be used for clustering the web services semantically based on the operations provided by the web services using multi-agent systems.
1.6 MOTIVATIONS

The basic infrastructure for web services serves the same purpose that CORBA and COM serve for traditional distributed computing infrastructures. However, there is an important distinction between the infrastructures for web services as opposed to traditional infrastructures. The inter-operability problem among web services is more challenging than the inter-operability between traditional distributed enterprises. One reason for this is that web services may be separated by firewalls and the semantics of data being communicated between them may not be uniform at the communicating ends. Another important distinction between web services and traditional distributed component models is that the binding between various web services is looser and can occur later than the binding between various components of a traditional distributed application.

Web services vision enables a dynamic world by allowing business processes to be modelled as web services, by providing a platform for hosting such web services, by defining the technical conventions that enable the interoperability between web services, and by defining a hub, or aggregation mechanism for web services.

The key differences between the traditional distributed computing models and web services infrastructure is that the message formats need standardization. This is in addition to standardizing the APIs needed to access functionality.
1.6.1 Issues and Challenges in Web Services Similarity Measures

Web service discovery is a process of finding the web service descriptions from the UDDI. The service providers register the services at the UDDI and the consumers can search for the services during web services discovery. The major issue here is to find the relevant web services among the large number of services. Another major issue is to discover the services in less time. One way to address the above issues is to cluster the available services. The existing approaches cluster available services using lattices based on the operations provided by the web services. The semantic information is not considered in these approaches.

In the present model, the issue of semantically clustering the web services in a lattice based operation is addressed. Lattice based clustering is achieved using Formal Concept Analysis (FCA). A concept lattice is feasible for small to medium size collections. The size of the concept lattice can grow exponentially with respect to the number of contexts. Concept lattices cluster web services based lattices using Formal Concept Analysis (FCA).

In earlier research, lattice structure for clustering web services was used and it was supported by Candidate Backup Services to ensure continuous functionality. A service lattice reveals the invisible relation between the services, easing the discovery and identifying the possible alternatives. The major drawback of the earlier works is that the semantic information is not considered during clustering of web services. In the present work, semantic information in lattices is considered as it is
easy to represent and also provides the hidden relation between the web services that are present.

The present research proposes a similarity measure that calculates the semantic similarity of the operations provided by the web services. The proposed similarity measure considers the harmonic mean of WordNet and Normalized Google Distance for calculating the similarity between the operations.

1.6.1.1 Current Challenges

The present research is oriented towards dealing effectively with some of the drawbacks of the earlier approaches and hence finds novel ways to make Web Services more efficient.

- The set of words cannot be changed or altered. Only the technical words should be specified so that the semantic information will be considered.

- The Concept Lattice is created based on keyword match rather than the Semantic Approach, the semantic information and the service lattice with QoS constraint is not considered.

- The Logistic Regression Model for Semantic Web service is not effective and appropriate for integrating individual similarity values obtained from various matching strategies on different description components.
• Though the semantic web services are described using various tuples, WordNet is mainly used for service description and also for the technical word representation for semantic process.

• The PSO based Clustering proved to outperform classical partitioning techniques as it avoids the problems of local optima stagnation, but a lot of computation is involved due to adopting PSO.

• In the Ontology based Semantic Web service clustering, the domain ontology should be as complete as possible and there is no unified algorithm for concept semantic similarity based on domain ontology.

• The Service Clustering is mainly for gaining the basic understanding of the cluster but it is not suitable for process similarity; hence in grouping of the clustered process in the form of Lattice, the process similarity cannot be achieved.

In summary, a research on web services clustering in future needs to address the above challenges with the attributes mentioned. Thus in this research, concept lattices are used for discovering web services. A semantic similarity measure for calculating the semantic similarity between the services has also been proposed.
1.6.1.2 Attributes

The web is an important part of people’s personal, professional and social life and thousands of services are becoming available online to support them. Since 2005, many efforts have been made to semantically describe web services and several models have been proposed towards this direction. Web Service Clustering is a process of grouping a set of similar services together based on the user request. The Clustering Technique alone is not sufficient to address the current challenges; an efficient model is required to process the services. Even though the semantic web service has been clustered based on Ontology, it has not been complete as there is no unified algorithm for concept semantic similarity on domain ontology. Hence an efficient model of Clustering is needed. In short, the challenges faced by the web services discovery have been listed above which mostly concentrate on the below attributes:

- Service Similarity / Service Matching
- Technical Word Representation
- Increase in response time
- Mismatching of services
- Performance of the Service Selection
- Service Relation
To summarize, the similarity measure for clustering using lattices has been based on string edit distance only. This poses the problem of not considering the semantic information while clustering services.

1.6.2 Need for Improvement in Web Services Similarity Measures

In earlier research, lattice structure for clustering web services was used and it was supported by Candidate Backup Services to ensure continuous functionality. A service lattice reveals the invisible relation between the services, easing the discovery and identifying the possible alternatives. Even though the semantic web service is clustered based on Ontology, it is far from perfect as there is no unified algorithm for concept semantic similarity on domain ontology. The Service Clustering is used mainly for gaining the basic understanding of the cluster but it is not suitable for process similarity; hence in grouping of the clustered process in the form of Lattice, the process similarity cannot be achieved. Hence an efficient model of Clustering is needed.

1.6.3 Issues and Challenges in Web Services Clustering

The related research work presented in the previous section clearly indicates the factors that affect the performance of the web service discovery. The area of web service discovery is developing as a conspicuous technology both in technical and business verticals and numerous research works are being developed in this domain, yet there are challenges to be addressed. The following discussion clearly identifies
the challenges in this domain with respect to the related works presented in the previous section.

- The web service developer or provider must use only the technical words specifying the functionality, in other words, the semantic information is not considered.

- Lattice based techniques for discovering web services with composite backups consider the operation similarity, but the QoS constraints are not satisfied.

- Trust-QoS based semantic service discovery method minimizes the discovery duration but the service does not execute any meaningful operation.

- A novel architecture for web services composition extends the standard web service composition model to suit exclusively the web services composition, which introduces too many new interactions thereby increasing the time.

- Ant-Inspired technique for composition and selection adapts the Ant Colony Optimization (ACO) technique for web services composition and selection without considering the QoS attributes and in the order of n cube.
• In discovery based user requirements, efficient web service discovery is guaranteed with web service analysis but testing is not considered.

• Semantic service discovery based on user preference cluster is a model which increases the service discovery recall ratio with the cost of increasing the average response time.

• The semantics based automated service discovery uses a semantic based service categorization and semantic enhancement of the service request, but the QoS parameters and the automatic service discovery are not considered.

• In web services discovery based on semantics and clustering, the services are discovered semantically, but more time is required during the pre-processing stage.

• The semantic service search engine identifies many semantic services during search, but the prototype does not support even pure WSDL and RESTful services.

• Web Services Discovery based on multi-criteria approach can discover the relevant services in less time with some QoS parameters, but the test set taken is very small to evaluate the proposed approach and also more QoS parameters can be considered.
• In the existing web service discovery system, the semantic relevance among the services is not considered, but the quality factors are considered.

• In web service discovery framework using OWL-S advertisements, the system has been evaluated to perform with high precision and recall against the typical query processing system, but the quality factors of the web services are not taken into consideration.

The challenges faced by the web services discovery listed above are mostly based on the factors that affect the performance of the web service discovery. The attributes of the challenges can be identified as follows:

• Difficulty in representing functionality

• QoS Constraints

• Increase in response time

• Increase in the pre-processing time

• Lack of support for certain services

• Insufficient test data

To summarize, any research in the area of Web Services Discovery in future needs to address the above challenges with the attributes mentioned. Thus in this research work, concept lattices for discovering web services are used in order to address those issues.
1.6.4 Need for Improvement in Web Services Clustering

There is a clear need for improvement as the existing approach doesn’t take semantic information into account while clustering web services. The existing web service discovery system does not consider the semantic relevance among the services, only the quality factors are considered. This is a huge drawback. The current situation also calls for improvement in representing functionality, response time, pre-processing time etc., Lattice based techniques for discovering web services with composite backups consider the operation similarity, but the QoS constraints are not satisfied. Considering the semantic information and the service lattice with QoS constraint is the answer to this problem.

1.7 OUR CONTRIBUTIONS

Based on the discussions so far, the Web Service Clustering needs methods and means for the QoS and other challenges identified. This research offers an enhanced model for UDDI registry with improved performance in terms of availability of the service information associated with UDDI, overall consistency of entries of the registries and service search time in the distributed environments. The attempt is to contribute the outcome to the domain of Semantic Web Service Discovery,

1. The main aim of the proposed research is to propose a framework to Semantically Cluster and Discover the Web Services using QoS based User Preferential Model.
2. The motivation is the inherent shortfalls of the existing models of webservices discovery models in the Web Service Computing environment.

In view of achieving the goals defined, the reported research is organized into several phases and the major objectives are listed as follows:

1. To Design a Web Service Clustering Model for appropriate segregation of Web Services using Lattice Concept.

2. To Formulate a User Preferential Model (UPM) to Bridge the Semantic Gap between the Service Request and the Service Discovered.

3. To Build a Semantic Service Discovery Model (SSDM) based on Clustering and UPM Models.

4. To develop a Test Bed to validate the performance of the proposed models using suitable Performance Assessment Factors.

1.8 ORGANIZATION OF THE THESIS

A lot of effort has gone into the organisation of the thesis so that arguments, hypotheses, illustrations and clarifications are coherent and lucid. The underlying view is that the work should be an important and strong link in the chain of research in the area concerned.
• Chapter 1 provides a broad overview of the Web Service Computing Model, Web Service Conceptual Stack and Web Service Discovery Model.

• Chapter 2 details the basic background information and the motivation for the research in the area of web services and concept lattices.

• Chapter 3 reviews Goals and Research Methodology, how the Web Service Computing Model can be used to enrich Conceptual Stack Mapping. The chapter introduces the fundamental ideas behind the web service similarity measures.

• Chapter 4, describes the concept lattices and their significance, and briefly outlines their adoption and the overall system design.

• Chapter 5 elucidates the experimentation and evaluation of the Concept Lattice Model.

• Chapter 6 presents the Semantic Web Service Discovery Model and its significance, and briefly outlines its adoption and the overall system design.

• Chapter 7 derives the experimentation and evaluation of the Semantic Web Service Discovery Model.

• Finally, Chapter 8 provides the concluding remarks of the work presented in this thesis and the future enhancements of the proposed line of research.