CHAPTER 1
INTRODUCTION

Increase in component based systems has created avenue for the development of Web Services, the inter-operable, platform independent and distributed components that can be used as a part of an application to perform a task. In general, Web Services in general provide the users with a single functionality.

Web services are computational components designed to build service oriented distributed systems (Zeng 2003). Growth of such architectures have led to a growth in the number of web services providing similar functionalities. However, they are differentiated in terms of quality levels provided by them. Customer’s satisfaction levels while using these services are determined solely by the Quality of Service (QoS) of the service (Zeng 2004), (Canfora 2005). However, it is not necessary to provide the best available service to every user. Performing this type of service assignments may not only underutilize the capabilities, they also tend to increase the cost for the user. Service requests are usually paired with the QoS requirements, and it is sufficient to allocate a service that satisfies the user’s requirements (Alrifai 2009), (Alrifai 2010). Service selection and orchestration are usually performed by keeping a single user in mind. However, the overload of web services is not considered. Other
issues in this section includes missing QoS requirements. The selection schemes are usually proposed by considering that the QoS requirements from the users are complete. In real time, it may not be possible.

It is known that the efficient working of any process requires a combination of many web services. Hence the process of web service composition is considered in this thesis. This helps in aggregating the web services such that it creates a chain of web services to accomplish a complete task. This in general is referred to as a workflow. Due to the increase in the number of web services, complexities in the process of selection have also increased. Further, each web service pertaining to a single process has different specifications and properties. Every web service concentrates on providing an efficient solution considering different quality attributes in concern. As a result, selecting web services based on the problem is considered the most inefficient way of building a workflow. The quality attributes specified by the consumer must match with the quality attributes concentrated in building up the web service.

1.1. COMPOSITE WEB SERVICES

Service Oriented Architecture (SOA) is a software design and software architecture design pattern based on discrete pieces of software providing application functionality as services to other applications and is referred to as Service-orientation. It is vendor, product or technology independent. Web services are used to provide a Service Oriented Architecture(SOA) that help in business process composition.
Service Oriented Architecture (SOA)s have become dominant in the recent years due to the fact that web services provide cost-efficient and better solutions for enterprise problems. The flexibility and standardizations in communication and the interfaces provided by Web Services act as a huge advantage for enabling the adoption of Web Services. This trend is accompanied by a huge increase in the number of Web Services available on the web service market. These web services are not only as simple as providing geo-locations, storage or tracking shipments, they also tend to perform complex functionalities such as online payments, credit scoring, customer profiling etc. Further complexities are also added to the Web Services enabling several complex business functionalities helping enterprise industries to a huge extent.

The importance of Service Oriented Architecture (SOA)s have increased rapidly in the recent years, requiring enterprise firms to implement their own services for flexibility and improved performance. However, the major question arises as to whether to develop the required services internally or externally. Both decisions have their own set of advantages and tradeoffs. Developing services internally is usually costly, however modifications and standardizations can be easily incorporated into the service according to the organization’s requirements. Several firms have currently developed their own web services and have also commercialized them by exposing and selling them to other organizations.

Usage of external services is cost effective, however, it might prove to be an overfit or an underfit, which needs to be compromised. Creating a custom web service usually meets all the required QoS standards, hence this thesis deals with providing solutions for organizations
that utilize external Web Services as a part of their architecture to derive solutions for their customers.

This section describes the basic properties of Web Services, their usage, major requirements for selecting and chaining Web Services to help provide a complete architecture.

1.1.1. Web Services

The W3C defines a Web service as a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically Web Service Definition Language (WSDL)). Other systems interact with the Web service in a manner prescribed by its description using Simple Object Access Protocol (SOAP) messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards (Web 2004). They are self-contained modular units that provide certain functionalities to perform a task or a part of a task. It provides a standard format in which applications could interact with other applications irrespective of their operating system, architecture, etc. The functionality of these Web Services should be described in a format that should be understood by all. This is carried out using WSDL (Web Service Description Language). This description helps you in the semantic determination of the web service by the user.

Web Services provide fine grained services, while clients require coarse grained, aggregated services. This mapping is performed by the process of web service composition. The process of combining the Web
Services in such a way that it performs sequences of a specific task is called web service composition. Due to the availability of numerous Web Services, and each satisfying specific QoS properties, an automated composition technique is much more advantageous and accurate compared to manual methods. Web service composition can be performed in two different ways:

- A simplistic low-level process modeling and execution languages (like WS-BPEL)
- A complex reasoning process, using high-level unambiguous description language for Web Services (like OWL-S)

Composition of Web Services can be performed using web service orchestration or web service choreography. Each process has its own positives and negatives. The working of orchestration and choreography is discussed below.

1.1.2. Web Service Selection

Development of service oriented applications often warrant several functionalities to be integrated in the single solution. These individual software pieces are to be identified and integrated. The required service can be accessed by using the service discovery registry. The major challenge is to identify the best set of services from the service repository containing a huge set of services. With such a huge list of candidate services, the developer must be skillful to select the best service satisfying their requirements in terms of both quality parameters and in terms of cost.
This has lead to the development of several automated mechanisms for performing effective service selection.

Service selection has become one of the major functionalities for building a business process. This holds good for any type of online application due to the fact that most of the application designers prefer to utilize the available services rather than build their own custom services. This decision could be considered uncomplicated if the number of Web Services available are minimal. However, due to the advancement in technologies and increased requirements, the number of Web Services available is huge and hence may not be solved by traditional selection methods. Further, the requirement of users does not come in terms of a single value. Instead, the demands come in terms of quality values for each attribute provided by the service. As a result, this becomes a Multi Criteria Decision Making (MCDM) problem. The requirement of any system solving this problem is that the result should satisfy all the demands of the user. Sometimes in case the rigid requirements are not met, the system should also be flexible to introduce tradeoffs in the selection process. Further, the processing time also proves to be a major constraint in such a system. A system that performs these processes faster and with best possible accuracy is the need for the current requirement scenario.

1.1.3. Web Service Orchestration

The appropriate selection of Web Services is usually followed by combining the Web Services in a particular order so as to provide a complete chain of process that completes a task. Web service orchestration and web service choreography techniques enable us to perform the
operation of chaining Web Services such that it can be used to complete a business process.

Orchestration usually refers to an executable inter-organizational process that is provided by one party (Mendling 2008; Matković 2012). The executable process has the flexibility to interact with both internal and external Web Services. Web service orchestration is usually viewed as a method that helps in the composition of multiple Web Services. Due to the fact that both orchestration and choreography are very similar to each other, several process are available that helps in deriving orchestrations from choreographies.

Orchestration in-general deals with business processes utilizing both internal and the external services to perform its task. The processes are usually controlled by agents representing centralized units of control.

1.1.4. Web Service Choreography

Choreography is an integration mechanism that describes message sequences between different parties, and the conditions and constraints under which messages are exchanged in an inter-organizational business process (Mendling 2008), (Matković 2012) (Kavantzas 2004). Message interactions between the Web Services are both local and global. Local interactions deals with communication within a single process, while global interactions take into consideration an overall point of view. Choreographies cannot be self executed (Decker 2009). Hence, it requires a co-ordination mechanism that helps in performing this process.
The major distinction between orchestration and choreography is that orchestration specifies the behavior of the participant in choreography. A set of rules are defined and their execution determines the process that is to be performed next. On execution of a rule, the process that is to be performed next is obtained. Orchestration assumes that a centralized control exists in the system, which governs the overall workflow of all the activities in the process. This centralized control works to perform the composition of a workflow utilizing the existing services.

Choreography operates on an assumption that there is no centralized control in the system. It is meant to be operated upon by peers in a virtual organization. They are used to verify if the operations are performed according to the specified rules. They are declarative and are executed during run-time. All the collaborators are required to use the same language to describe the business protocol (choreography). One of the most popular languages for service choreography is the Web Services Choreography Description Language (WS-CDL). It defines a multi-party contract in the domain of Web Services. The major requirement of a WS-CDL is that they are required to completely hide the background implementation details and are expected to be flexible in terms of operating parties and the software used to perform the operations.

Choreography definitions written using WS-CDL are more robust and provide excellent validation measures. They also ensure service interoperability, hence making the multi-party contract effective. It provides a global view of the messaging activity irrespective of the implementation nuances. Hence individual respondents generate code based upon the role specific code skeleton.
1.2. THESIS OUTLINE

The remainder of this theses is organized as follows: **Chapter 2** presents the literature review discussing prior works under the domain of web service selection and orchestration, **Chapter 3** presents the objective of the present work and research framework, **Chapters 4** presents a comprehensive survey of techniques for web service choreography, orchestration and workflow building, **Chapter 5** presents a user preference based service selection process for Web Services, **Chapter 6** presents a discrete, modified PSO based service selection and orchestration scheme, **Chapter 7** presents catfish PSO, which is used to overcome the problem of local optima in PSO and **Chapter 8** concludes the thesis.