CHAPTER 1

INTRODUCTION TO SERVICE ORIENTED ARCHITECTURE

1.1 Introduction

In the times immemorial, we treated p2p (point to point) model of communication that concentrated from one to one communication where each minutest point could be treated either as a node, client or software component. After this our ideas took path towards hub and spoke and later the ESB (Enterprise Service Bus). Ultimately we came around with the watch word called ‘Service’. In the field of information technology, complexity is considered a reality of existence. By and large our attention was diverted towards the functional testing of a web service that is considered a kind of black box testing which maintains specification based test cases in the element of the software under the test (web service). The functions are constantly tested by giving them with the input and finally the examination of the output, least bothering the internal program structure. Functional Testing methodology is explained elaborately in [1]. The major confront is to deal through complex nature while making brand new applications, substituting current ones, and sticking to the maintenance and development requests. When a common programming interface and interoperability protocol is to be used at all applications it becomes much simpler for the IT reducing the complexity and the present functionality could also be reused with much ease. This is possible only if the existing IT infrastructure is reorganized. The IT world can consider it a boom when the service-oriented development is deployed by the use of Service-Oriented Architecture (SOA) [2]. This further opens the doors to latest tactical solutions which include:

- Integration of varied application.
- Business processes which are automated.
- Accessing of multi-channel to applications that involves mobile and fixed devices.
The part of composing of services across the pieces of software which are dissimilar is facilitated by an SOA. Be it older one or novel one, even organizational, whether business wide or inter-business, mainframe, and even the middle-tier either mobile or personal computer, to reorganize the processes of information technology and avoid the obstacles to the enhancements in the environment of information technology. These composite solutions of an application are at hand’s reach just the reason being the widespread receipt of the well-defined web services and unbelievable switch over supremacy of an SOA. The WSDL (Web Services Description Language) became a genuine and standard and basic programming interface in the process of accessing any application and it also understood that SOAP hold its hand high as a standard interoperability protocol in connecting of any application to one another. It is clear that these standards gave remarkable opening which are further followed by a plethora of additional web services specifications which could define the aspects of security, reliability, orchestration, transactions and managing the metadata in order to meet further pre requisites for features of enterprise service quality. Above all, the so called web services standardizes as the finest stage over which the SOA could be built and this can be termed as the next-generation IT infrastructure. Web services architecture is further explained in [3].

Here the discussion is about the manner of enabling applications to perform the work rather transparently by cooperating with one another across diverse computer-system/platform/operating-system/environments of different languages. It is considered a concept and a group of internet standards that were designed to enable and to perform different Web-based program-to-program communications. Hence, we named the same as Web services. In a way, SOA is a manner of building/creating applications and systems in a better flexible format, and by doing so, it is clear that IT has become suppler in adhering to the changing trends of the business needs. SOA has spread into the very vitals of the human works and can even turn out as a flamboyant signal of the future generation. To run the same in varied operating environments, coded in several languages, with differed programming interfaces and protocols to the accurate levels in technology. SOA can be treated as a concept to deliver the information as a nitrous in business modernization.
1.2 Importance of SOA

As a matter of fact, whether to go for SOA or to decide not to go for SOA is no more a question anymore. It is finally 'when to go for SOA?'. By the rapidly matured SOA implementations and awareness of the involved challenges and advantages, there is a rampant use of SOA in the path of the developing business enterprises. The importance of service-oriented architecture (SOA) and why it is one kind of valuable picks for the architecture of software to build software based on web and systems is explained in [4].

The department of information technology carries out their work with ease because of the SOA. It cleared the path for a shift from an application point of view to a process view. In the modern times the IT world enjoys a full length freedom where they can merge services of business from various applications in order to convey a genuine uninterrupted support to business processes. As the integration method of SOA generally the Web Services enable a loosely coupled components, the departments of IT can go for changing their applications without disturbing other applications. It has been observed that nearby in attendance is a rich shortage of hard testing and composite methodologies designed specially and particularly for SOA applications, though it has been progressively implemented as together top down and sometimes bottom up. It is better to validate and verify the applications that depend on concepts of SOA, and it is necessary to adopt new approaches and methodologies.

I think it is highly difficult to say that web services are equal to web applications; it is an application or a service which is accessed enroute web over a network, for example internet or an intranet. Though the concept and essence of SOA and well-defined services are varied aspects to understand SOA, the web services are the ideal one which follows different standards. In the creation of software applications, SOA is the architectural model which uses the available services of a network for example web. It acts as a catalyst in promoting the loose coupling among the components of the software that they can be reused. As far as SOA is concerned, construction of web applications is based on services. A service can be understood as an execution of business functionality which is well framed. These services can be utilized by the clients in various applications or else in the business processes.
1.3 Features:

1.3.1 Reusability

It is the need of the hour to know and understand where and how these applications and systems stroll to commune with one another. It is so risk taking to analyze and the effort of development involved chiefly depends on tying these pieces together and it is also a lengthy process. The organizations of IT face a great deal of challenge that in general comes across while trying to put together all these threaded applications.

1.3.2 Interoperability

The communication between clients and loosely-coupled services is the vision of SOA and is called interoperability. The most generally suggested technology for the concept interoperability in SOA is implemented by Web services is further discussed in [5]. This aim can be achieved if only the services and clients maintain a standard manner of communicating with each other i.e.; consistency across platforms, systems and languages interoperating set-up among components. This in-fact means what exactly a web service is. In a way, the web services are a blend of matured group of protocols and technologies which are accepted and used widely and are system, platform and language independent. Web Services Interoperability Organization – this is an organization usually encourages interoperability concept of web services, recognizes and acknowledges a specific group of different web services technologies when applied in various platforms and systems. An extensive back up in the services of internet and computer business division is maintained by WS-I profile which almost guarantees interoperability of any service which obey the rules of the profile.

1.3.3 Scalability

Since the concept of services in essence to SOA are coupled loosely, the applications utilizing the said services are inclined to balance rather easily and no doubt even more often effortlessly than the predefined applications which are in an additional and simple tightly-coupled abstract environment. The reliability between the client and a well-defined service in an environment of tightly coupled is bounded; where attempt of modeling was more difficult in the form of an
application using these services balance up to handling more users. Services, for instance in a web services-based SOA appear to be coarse-grained, document-oriented, and asynchronous and interactive.

1.3.4 Flexibility

SOA is a style of IT which supports the architectural conversion of business into a linked set of services, or tasks of business, etc. The services that are loosely-coupled are bendier when compared to tightly-coupled applications. In view of tightly-coupled, various components of an application are bound very tightly to each other and shares the semantics, libraries, and quite often shares resources. This further worsens the application become difficult to evolve and grow at an infuriated tempo to meet the requirements of the business. The SOA provides flexibility to the loosely coupled, document-based, asynchronous environment of services and it in future becomes easy to evolve with the varying business configuration.

1.4 Impact Investigation of Service Oriented Architecture

In the designing and building of the systems to provide support to a lively business environment, SOA is considered a variable approach. The competence of adding latest capabilities and objectives enhanced can serve to reduce the expansion expenses and can more or less remove established development cycles. A service may be widely used throughout the enterprise by many applications. The enterprise as a whole can make use of a service using widely by many applications. The benefit at the first phase of an SOA is the applications which can be built by both combining loosely coupled and interoperable services; the truth being that a service might be utilized at a wider range all over the enterprise by so many applications can show the way to the further exposures for IT infrastructure in the following manner:

- The time of response is very poor and not particular in the applications of a business process.
- The different levels of service are missed for vital business functions.
- Nonconformity with the area of a particular sector and public set of laws and norms.
- Insufficient management of the services and resources.
With the help of SOA, if needed to alter a business policy or a plan, just change it in one position for the similar service is used at all places. This gives consistency in the organization. Changing the role of the IT infrastructure with the SOA, the foci need to be shifted towards management of services that support business processes and, thereafter ensuing in optimization of business and customs. As far as the middleware integration development is concerned, SOA places gleam performance and necessities of scalability. When it comes to handling of messages and the importance of the enterprise service bus (ESB); In our way an ESB is an architectural element which can be intended and deployed in a way that provides a similar process to the business environment. This can be incorporated in several ways, such as with classical messaging, EAI as explained in [6,7] etc; or by use of the components which are platform-specific for example: the service integration buses in J2EE systems as in Web Sphere Application Server (WSAS).

An approach, unique when transactions are processed determines the significant style of SOA. With the good old traditional applications, transporting and translating of the messages occurs in the application layer till the end of entire transactions. The transaction gets started in the application, with SOA. Nevertheless, the messaging has been dropped off the application layer and presently extends into the infrastructure of the middleware. With SOA, the application of the business process gets initiation to the transaction and the ESB provides the messaging, transformation of the data and intellectual routing.

The following are the functions of the ESB:

- All the way through xml messaging only the routing among the service components is possible.
- The transport protocols may be converted among service provider and the requestor.
- The formats of the message can be transformed between the service provider and requestor.

The ESB [7,8] extends the services to consumers, providers and even to the applications independent and unique from each other. ESB is critical part of SOA infrastructure. At the beginning, the architecture of ESB just transports messages but at a later stage of implementation
further intricate routing and the transformation of messages are included. Generally, ESB’s are used to connect Applications to each other through a port as explained in [8]. In the process of attaining uninterrupted performance, ESB is a very vital component and the implementation of the same is also serious in several ways. However, it is very important to reuse the standard infrastructure services that ensure compatibility and reliability. Hence, in a best-practice-based implementation, the intermediaries should utilize the standard Java Web services SOAP-handling standard: JAX-RPC. SOA makes processes more able and as we start new services, the process flow of the business can be effortlessly modified to arrive at new capacities. As an addition to this, since the services are upgraded or even sometimes replaced, the ESB automatically sends a query request from the server side for accessing the new and latest version of the service. The SOA governance helps in assuring the services those are reused effectively across the enterprise. A considerable performance is yielded by SOA that will show an impact on both network and the components of infrastructure. A Framework is proposed in [9] which explain impact analysis across different layers of SOA based system.

1.5 Impacts of SOA on performance and testing

A messaging structure named eXtensible Markup Language (XML) [10] is used by SOA. It is a wordy message format that can have impact on the network performance as well as the components of the infrastructure responsible for the process of message format. Most likely, SOA services are distributed all through the enterprise that can result in numerous communications across the network. It is necessary to prioritize, to analyze, and to sneak a look at the right level of granularity, the moment appropriate software is identified for SOA. Surely, granularity can have an impact on both the performance and testing as it encounters the notion of interoperability. For the services that are extensively used, improper planning of infrastructure and design may cause overload and it can be a failure in the network systems. This further result in deprived performance, slow down the response time and even a possible downtime for many applications and users. Let us not forget our main aim - to make sure that the time taken for the complete process of transaction in the environment of SOA shall not become slow or even less reliable than the environment a traditional application. This provides a lively creation of new applications through redefining business.
The process of refactoring is carried out in the form of steps. After the completion of each small step, we remain in working system unchanged functionality. Practitioners quite often interleave bug fixes and feature add-ons among the sequences of steps. Hence, it is understood that refactoring doesn't prevent changing the functionality, further it maintains as a special activity by rearranging the code which has various aspects in rearranging the code on short run. A useful support is provided by refactoring, when renaming or moving artifacts through the implementation of changes across a blend of artifacts necessarily not changing the performance of the application or even troublesome dependencies. If we try to bring a modification by using refactoring, the dependencies get updated by the program automatically across the structure of the application and a section of its own. Till its withdrawal from its inception, a service should persistently undergo a kind of change either in the logic itself or in its underlying implementation. And the said change is either qualified to the raising demands of its consumers of the service, that it is not possible to hold anymore, or to the requirements improvement of the underlying heritage or old systems. It is not a point if the source of the change is either from internal or external to the service architecture, there is a dire need of addressing the shift/change not breaking any dependencies between the consumers and the contract of the service. This shift/change is best executed by the service application. Refactoring depends on the principles and techniques observed and governed by the object-oriented design standard. In [11] a J2EE application is refactored into JBI environment, this aspect of the pattern encourages low incremental changes so as to satisfy on the whole objectives at compound level. By advocating the utility of multiple contracts the applications of the concurrent contract pattern may be proved quite useful. The pattern of service refactoring tends to bring a change functional context service. This in turn may need to move a particular service capacity away from the service contract into a fresh or even an existing service.

In a way, it is possible to import Web Service Description Language (WSDL) files with inline elements. With the use of refactoring, the inline elements can be extracted into separate files. Just by right-click on the object in the business integration view, almost all the refactoring operations can be initiated. The elements those do not appear in the business integration view, such elements can also be refactored.
1.7 What makes SOA testing different?

Though the answer varies in different dimensions, it maintains the bottom line as liveliness and flexibility. It is true to say what makes SOA so attractive and a business friendly IT standard being the same why it is necessary for a different testing approach in SOA Implementations. While dealing with the testing of SOA applications, it is needed to look ahead of functionality and performance testing. The testing of SOA needs interface testing and service testing those could bring all together varied abstract systems and configured platforms in lieu with other security and performance linked aspects.

According to James McGovern, Oliver Sims, Ashish Jain and Mark Little [13] another major challenge to be handled in testing of SOA is a fair accessibility of the background along by means of the reliant essential core services and even applications. As an instance, while to make up a business process, an SOA in accordance with implementation may carry close two or even extra self-governing inner services. The availability of the same turns out very important at the time of conducting integration formal testing in slices and even through end-to-end testing of the predefined business process.

This chapter starts with a short introduction to an abstract SOA and further frames an identity to the testing of a SOA basic implementation precisely in an efficient and trustworthy way. A clear description is given about different categories of testing, suggested test approach and information about the available tools in the global market those are useful to complement the complete testing strategy.

1.8 Key vocabulary of SOA

A service can be understood as a business task or even sometimes as process task which at times be repeated. Services are utilized so as to put in a nutshell the serviceable pieces of an advanced application through giving a practical interface which is at the best described and implemented independently. Services often are stimulated by any other services or even applications.
Service orientation usually can be defined as the process of integrating all the commercial applications and most commonly used procedures as services which are linked.

Service-oriented architecture (SOA) [2,3,13,14] is understood in different ways by different people basing on the role of a person and context. From the business point of view, SOA maintain a group of business services which are composed to capture the design details of the business needed by the enterprise rendering internally and also its partners and customers. In the view of architecture, SOA appears as an architectural method that defends and wires service orientation. At the level of implementation, SOA has to be fulfilled by the use of a values based organization, replica of programming and core technologies like web services. As of the operational point of view, SOA involves a bunch of contracts among service consumers and its renowned providers who spell out the service quality and also reporting of the crucial marketable and various metrics of IT.

A composite type of application is a bunch of integrated services and related to hold up a commercial method constructed on SOA.

1.9 Vital Components of SOA

According to Thomas Erl [12] SOA comprises of the following components -

- Service provider
- Service consumer
- Service registry

Each and every element can perform and can act as equally as the other two components. For example - suppose a service provider requires some extra information which can be obtained from the other service, it performs its task as if it is a consumer. The operations that each component can perform are shown in Figure 1.1.
A service is created by the service provider, and in very few cases its interface is published and grabs the data onto a service registry. It is sole responsibility of the registry of service to make the interface abstract and performance admittance data obtainable to all the consumers. The consumer identifies and selects the entries in the registry of service and later bind to provider so as to call upon the said service.

1.10 Testing Methodology of SOA

Most of the organizations these days are looking at SOA because component reuse is the characteristic property provided by it. The test team of SOA should not only

1.10.1 Traditional Test Approach

Conventional testing of software had all ears on testing at code level and it gets around through architectures of web services and of type distributed. Further testing of business logic with the user interface of an application has been introduced by web application testing, which is shown serious while customizing newer results. By SOA, the necessity to check the sense logic of the business that exists; nevertheless, most of the SOA based services cannot possess an interface of user that are considered as the latest challenges to the test group or organization.
The following are considered to be the challenges of SOA testing [16]:

- The services those do not maintain user interface.
- Within services we can expect business logic which is data driven.
- Outside services to the group or organization.
- It has an imperative quality of service software in order to support the reuse and facilitating business liveliness. Those services having identified bugs and issues of quality cannot be used again by any of the teams that do development. A boost in testing actions and check resources is significantly necessary at a level of service.
- Foreseeing the future utility of abstract services to support with core ability to perform, stress, load and performance to scale.
- While the SOA evolve the testing at security level maintains a high range of concern and profile within the test approach of organization.

As far as SOA is concerned, services depend on varied technologies. We cannot go further for the test of an advanced application which gets developed by an integrated assembly, like an individual project basing on one application server and its delivery performing through a consistent browser interface. The capacity of stringing several groups of components together to form a business process need unhindered thoughts of an architect's view, and planning of the test and agenda of complexities through a tester's view.

In SOA, the logic of an application lies in mid layer, performing the operation inside various technologies, staying the department in exterior and sometimes exterior to company also.

It is a known fact that, in order to test SOA, we have move far ahead of testing a user interface or browser screen. The role of testing and monitoring a service oriented system is explained in detail in [16]. Web Services are of components for a good lot of Service Oriented Architectures, except when you test only the web services, it is not necessary to test the whole heap of technologies that creates the application. As a matter of fact, a lot number of legal SOA applications which are perfect appear in business logic outside the web services entirely.
Reviewing and enhancing their present test methodology is required for the organization in order to address the above said challenges. The vendors of various test tools by now have familiar with the latest challenges to test SOA and have already urbanized a brand new variety of tools in helping the organization to prepare a strategy, administer and to automate the functioning, security and performance testing of SOA.

1.10.2 Service-level Testing

Service testing becomes very important and significant test phase in the test approach of SOA. Now-a-days most of the organizations build a Web service or program that performs a partial unit testing and enhances its delivery to the phase of integration testing that allows the testing team to go for an evaluation of the quality. The reuse of the service is demanded after each and every service is dispatched from the very beginning phase of testing by providing a complete quality report and also an assurance.

In this phase of testing, the following test and quality activities recommended:

❖ Casual code reviews by peers, to make sure that it complies with the standards of the organization and identifying the potentiality of interoperability, performance and security measures and defects.
❖ Functional, performance and security regression groups in executing against the service. For this to happen, it needs the support of test tools which are automated and even the growth of classy stubs and harnesses.
❖ Entry and exit criteria of quality are not just for defining in this phase of testing but also achieved before the service is delivered to the next phase of testing.

Testing at the Service Level has to make sure that a service means just not to meet the necessities of the existing project, and the important thing is that it still meets the operational and business requirements of another process which use these services. In [17] a case study is illustrated to show how service level testing is done in detail.
1.10.3 Functional Testing

Functional sometimes called Black Box testing determines whether a component, an abstract service or the entire system operates to the requirement not referring to the in-house technological blueprint. The chief inputs to the functional design of test cases are the requirements of business and the higher level of technical design definitions. There are totally five steps involved as explained in [18], they are:

1. Identification the functions that are related to the performance of software.
2. Depending upon the function's specifications, creating the input data.
3. Depending upon the specifications of function, resolving output.
4. The test case execution.
5. The comparison of expected and actual outputs.

1.10.4 Functional testing based on requirements

Usually, functional testing is the initial test phase which a test organization is responsible at the time of a given discharge. The function test based on requirements is considered one approach to integration test which is an efficient and dominant testing approach that reduces significantly the total number of unnoticed defects got released into production. The argument being that a well-framed group of functional requirements gives the test designers a concrete base to design a test case.

Function Test – What does it mean?
The main aim of function test is to make a judgment of quality of the business components of the system. These tests are to verify whether the system behaves properly in-lieu with the business point of view and the functions in according to the requirements, models which are used to specify the application. The function test should establish whether each business event or component - performs as per the specifications, correctly responds in all conditions those may be presented by incoming data is correctly moved from one business event to the next and those business events gets initiation in the order necessary to adhere the business objectives of the system.
What is a Requirement?
A requirement is an ability or function which has to be delivered by a system component or even components. A functional requirement is a type of business behavior as observed by an external user of the system.

The test cycle for function test
The test cycle to be effective, it need to have a defined group of deliverables and processes. The necessary deliverables required for the function test are: test planning, functional decomposition, verification of requirements, test case design, execution of test case and analysis of coverage. What are the processes that are applicable to a given testing situation are depends on the resources available. For example, time, people raw materials etc; and the directive of the test organization.

Test Planning
While planning the test guide with support from the test team, it certainly demands the schedule, scope and deliverables needed to run the function test cycle. The test lead brings out a test plan and a test schedule and further these undertake numerous revisions in the process of the testing cycle.

Partitioning - Functional Decomposition
Functional decomposition is also called partitioning can be the breaking down of a system into components. Some other group among the organizations may handle the responsibility of the functional decomposition of the system. In spite of that the testing organization has to review this deliverable for totality before taking it into the test organization. Suppose the partition is not defined or even thought to be inadequate, the responsibility has to be taken by the testing organization for partition maintenance and creation. There are quite a lot of marketable products at our hand's reach to support in the functional decomposition of a system and the casual release of the functional partitions.

Requirements Verification
It is often said that requirement verification or definition is the weakest deliverable in the development process of software. Most of the development shops directly go from concept of
software to specification of function or at times to code not having even first round deliverables. The Testing Organization has to perform all these prerequisites to continue with function test. If the development team doesn’t deliver the requirements necessary for verification by the testing team then the test team has to build a set of requirements on its own and in turn need to catalog them under the suitable functional partition.

**Designing test cases**

The test designer causes the test case to validate the performance of the product as per the requirements. Further, the test cases are to be catalogued with the suitable functional partition and to be traced to the requirements necessary to be tested.

**1.11 Insights**

- Web service is architecture of distributed computing which forms the similarities by using loosely coupled core applications while provides service for one another. The web services element or feature a newly different way to perform communications between programs which allow applications to be in touch with each other, no matter which application language is used to create the application and even what sort of systems platform and in service background is under utilization.
- The platform elements of Web services are UDDI, WSDL, and SOAP. These are explained in detail in [3,19].
- Most of the complete Web services implementations involve application development environments, development of portals, managing business processes and even more and in a way all quoted above is extremely incorporated with Web services protocols and registry services.
- There are so many attributes related to the architecture of type distributed computing, they are Common Object Request Broker (CORBA) [20], Advanced Program-to-Program Communications (APPC) [21], Electronic Data Interchange (EDI), and more.
- The other manner of defining what exactly the web services do is: publish, find, and bind,
A different way to understand how Web services work is given here. There are three functions that web services directories and protocols basically supply: to publish, to find, and to bind. In order to use the web services, the applications strived are as follows:

- To publish in a directory of UDDI.
- Find – using protocols of WSDL you can locate applications of web services and
- Bind – a link is established between two applications, by which a service can be delivered using protocols of SOAP.

In my opinion, after considering all the points, it is understood that web services defined are a novel set of mixed-platform communications between program line to program line which allow the applications which are loosely coupled to locate each other with ease, to establish parameters effortlessly and vigorously that facilitate related or unrelated programs to go hard in hard further enabling the same to communicate in an automated and unattended style on the internet. The services of the web are applications that utilize registry and different standards of communication to perform together in a way dynamic and efficient where each application supports messaging or computational services to each other. Further, they make use of an XML format for the presentation of data. They also make use of web services standards to find applications like UDDI, give and take of how to drive and obtain the information which is a WSDL, interaction session binding like SOAP, and finally transferring this data on the internet which is HTTP. Fundamental technologies used in web services and Web service architecture is explained in [22]. Web services are no more a novel concept. From the times immemorial, the designed architectures to enable communication to communication platform have been present. For the reason that the consensus of the industry about how to present the data so much so that it can be manipulated and interpreted and even about the way of communicating the data with the use of internet. These two obstacles to cross-platform program-to-program communications have been conquered now. In regard to the seller promise from almost all the industry's most important suppliers of software and hardware provide an assurance that the services of web, by now become the widely established standard way to the varied applications to communicate via unrelated platforms.
The notion in general, behind program-to-program communications is to permit each application to effort collaboratively with the other on a communication network. In order to make this possible, applications have to be able to:

- **Distribute data:** This becomes mandatory that content in common is in use between each application. The examples of the same collected from the past include several architectures and makes an attempt to create industry standard forms with the use Electronic Document Interchange (EDI) formats.

- **Start Calls:** In general, calls are based on verbs, for example - send, receive, open, close etc and other activities which enable applications to share or distribute information and files. These calls get implemented with the use of a general program enabled interface called Application Program Interface (API), which basically consist verbs the applications use in communicating with each other.

- **Communicate on a network:** It is necessary that the Program be able to send and receive the data on a network which is common that involves a common communication protocol like SNA, TCP/IP etc.

- **Have an XML environment:** XML is highly important in the web services for it provides the wrapper in where understandable and calculating data can be accepted between contrasting applications.

- **Use Web services mechanisms:** When comparing to other communication architectures like EDI, COM and CORBA web services mechanisms like WSDL, UDDI and SOAP appear as young and immature from a perspective of security, manageability, and reliability. However, may premature adopters still use other communication architectures to send XML data until the latest web services standards mature further.
The specification Web Services Reliable Messaging (WSRM) allows two different systems to send messages between one another consistently with an aim to ensure that messages are transferred appropriately from sender to receiver. Reliable Messaging is very complex, but there are also possibilities of brooding over WSRM that provides a equal range of guarantee for XML messaging which a JMS system provides in the world of java. There exists first one key difference that is JMS which is a standard API with innumerable implementations and protocols beneath that. WSRM is opposite as it is a standard protocol without API of its own self. As a substitute it composes with accessible systems based on SOAP.

Logically thinking, there are two agents in WSRM - the RM Source which is RMS and the RM Destination which is RMD. There may be under implementation by one or more handlers in any known SOAP heap. An analysis on Reliable messaging and its importance is explained in detail in [23]. RMS and RMD are explained below:

**The RM Source:**
- Creation of requests and extinction of the consistency/reliability contract.
- Adding headers of consistency and reliability into messages.
- Sending messages again and again if required.

**The RM Destination:**
- Creation and termination of reliability is possible in responding to requests.
- Messages can be established and accredited.
- Replica messages can be dropped.
- In anticipation of lost messages arrival holding messages which are outdated.

It is very important to avoid the confusion of the source and destination with the client in terms of requester and server as provider. In a collaborative reliable setup; there is an RMS and an RMD in the client and even in the server. In [24], the most basic message exchange pattern used in web services is request – response where parties communicate simply with each other is
explained. In WSRM we add the delivery specification of the messages to basic request – response pattern to ensure reliability.

1.12.1 Take a quick look

By and large, it is all about deliberate way of making applications work clearly and considerately with one another across varied computer operating system, language environments. It is a blend of concept and a set of internet standards designed to facilitate interoperability, hence we named it web services.

Web services are considered architecture of distributed computing which makes the use of applications which are loosely coupled and divergent to tightly coupled applications in order to make the applications to communicate. Finally, the concept of tight coupling affects how the information systems work in the time to come.

1.12.2 Why it is unique?

At the end, this encapsulates the complexity in writing the applications of type program to program being translucent and hence simplifies the work of programmers. In the long run, programmers need to focus at length on writing constructive applications and less on making those applications go hand in hand with underlying communications of the system.

1.13 Conclusion

In this chapter we have presented the history and importance of SOA, in which the concept of web services play a vital role in different areas and also talks about reliable messaging. The concept of functional testing is also discussed which concentrate on designing test cases and also about the key terminology and vital components of SOA which enable the user to understand the background of SOA. Finally, how messages in web services transferred in two applications is discussed.