Chapter 1: Introduction

In today's world, computers are common place with variety software and large number of applications available readily for variety of tasks. In the past, though, this scenario was different; the computers were used mainly for and by the scientists for complex calculations. Very soon, the potential of computers was realized by the businesses / people. As a result of which, along with the hardware developments, the software development saw a tremendous change and consequently user friendly software and hardware were developed. The requirements needed to be captured and programmed.

Different languages and various development methodologies assisted and aided the development and documentation of software. The networking of computers leads to the different system architectures. Also, the businesses expanded themselves to be visible globally. All this lead to the generation and exchange of large amount of data which gave rise to commensurate technology of storage and distribution of these data. The storage and distribution of data becoming very important due to which the demand for reliable hardware and the customized software increased. The ever developing and demanding world of IT along with the dynamic businesses has seen significant changes in the way applications are developed, deployed and maintained. The evolution in developing and maintaining large and complex applications made it imperative to develop sophisticated technology of computing and storing and distributing the data.

1.1 Need for flexibility

According to John Charles Olamendy in 'Strategies for Managing Computer Software Upgrades' - to be competitive, companies have to be agile. They constantly had (have) to adapt their organization and environment to increasingly frequent technological changes, mergers and acquisitions. This necessitated the development of IT systems which are integrated and reactive, and therefore solutions need a loosely coupled integration, with the

possibility to progressively improve migration of legacy applications to newer platforms. Business processes are extremely complex because of the variety of technologies in the IT infrastructure, such as the operating system, application and enterprise information systems. Integrating such heterogeneous technologies is very costly, time-consuming and very complex.

The businesses are a witness to the ever expanding dynamic global market. Adapting to such a situation is crucial and at the same time difficult. Large amount of money is spent in developing, maintaining and providing security to the systems or applications and data thereof. Business process changes need to be quickly reflected in the systems which support these processes. Nowadays there is a great effort for Enterprise Applications Integration which involves a large number of companies and developers.

1.2 Current Software Scenario

With the world becoming a global village, the demand for information across the world increases tremendously. Information systems need to evolve so as to keep pace with the ever increasing software systems. The demand in software systems and the revolution and evolution of software tools and methodologies has been excellently given by Terry Woods as shown in Figure 1.1. Development in network and database technology, organizational and economic growth, interconnection of existing databases, incremental growth, reliability and availability and reduced communication overhead – all lead to the existing IT systems of today.

The relatively new Service Oriented Architecture (SOA) methodology is a recent advancement in the IT sector to meet the ever changing situation as regards the information generation and dissemination in the IT sector and businesses which reflects on storage, processing and communication of relevant data. SOA is based on the concept of services. It is a software design that is independent of product or technology. SOA is a vast area and its implementations are different as per client need or project requirement.

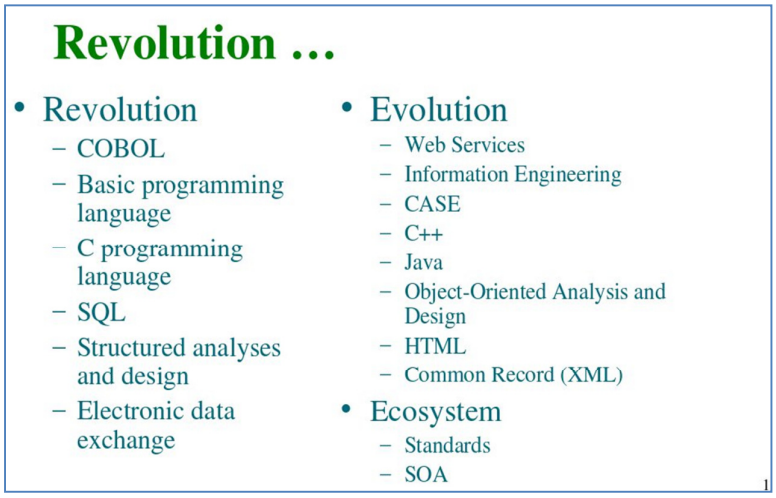

Figure 1.1 Revolution vs Evolution

Services are available at different layers like software as a service, platform as a service, infrastructure as a service etc, which has already spread into the IT industry. In the article titled “Software-as-a-Service: The Spark That Will Change Software Engineering?” Gartner’s Norton mentions that the transition to SaaS-based architectures is still in its early phase though by 2010, 15 percent of large companies would start projects replacing their ERP backbone with a SaaS offering.

SOA, promotes the idea that IT applications are service providers and consumers acting together to support the overall business goals of an organization. This new approach to enterprise architecture as posted by John Charles Olamendy is a loosely-coupled solution in which software components are very independent. He also provides an overview of how .NET supports the goals and principles of SOA. The main approach of .NET is explained along with the concepts of SOA and how .NET and the technologies associated fit into the strategy of SOA.

The evolution in software system development is excellently supported by the development in the Data Storage systems. The databases have evolved from flat files to the NoSQL databases of today. The distributed databases are also used by number of organizations to handle the vast volume of data distributed over the network and transparently accessed by the system components. The technical data storage details like the database management component, the

data communication component, the data dictionary and the distributed database component for the administration of distributed database, make the distributed database as it is today. In short, the Information Technology scenario in today's world is majorly concerned with the availability and accessibility of data through systems that are flexible and adaptable to the changing business environment and the customer demands.

1.3 Service Oriented Architecture

SOA has evolved from the use of DCOM or Object Request Brokers based on the CORBA specification. It is essentially a collection of services. These services communicate with each other. The communication can involve either simple data passing or it could involve two or more services coordinating some activity. Some means of connecting services to each other is needed. According to Thomas Erl, SOA is a software design and software architecture design pattern based on distinct pieces of software providing application functionality as services to other applications. It can also be interpreted as a flexible set of design principles used during the phases of systems development and integration in software development process. A software system based on SOA will package functionality as a suite of inter-operable services that can be used within multiple separate systems from several business domains. It is an evolution of distributed computing based on the request / reply design paradigm for synchronous and asynchronous applications.

1.3.1 Need for SOA

The reality in business enterprises as explained by Raghu R. Kodali, is that IT infrastructure is usually heterogeneous across operating systems, applications, system software, and application infrastructure. Some of the existing applications are used to run current business processes, so starting from scratch to build new infrastructure is not an option. Enterprises need to quickly respond to business changes with agility; leverage existing investments in applications and application infrastructure to address newer business requirements; support new channels of interactions with customers, partners, and suppliers; and feature an architecture that supports organic

business. SOA with its loosely coupled nature allows enterprises to plug in new services or upgrade existing services in a granular fashion to address the new business requirements, provides the option to make the services consumable across different channels, and exposes the existing enterprise and legacy applications as services, thereby safeguarding existing IT infrastructure investments.

1.3.2 Concept

An application's generic business logic or individual functions as mentioned by Raghu R. Kodali are modularized and presented as services for consumer / client applications as shown in Figure 1.2. The key to these services is their loosely coupled nature; i.e., the service interface is independent of the implementation. Application developers or system integrators can build applications by composing one or more services without knowing the services' underlying implementations. For example, a service can be implemented either in .Net or J2EE, and the application consuming the service can be on a different platform or language.

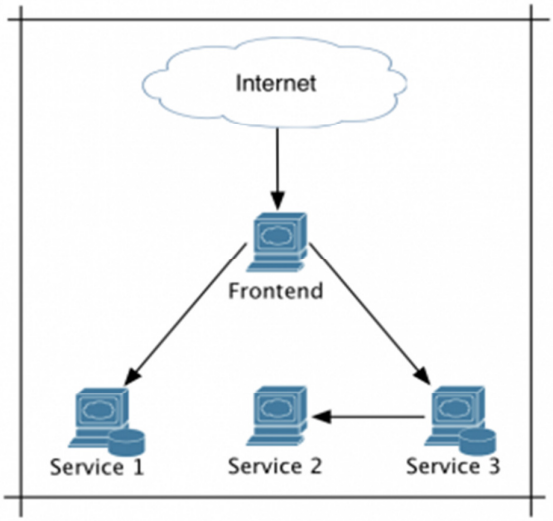

Figure 1.2 Simple SOA where services are called or communicate with each other.

SOA also generally provides a way for consumers of services, such as web-based applications, to be aware of available SOA-based services. For example, several disparate departments within a company may develop and deploy SOA services in different implementation languages; their respective clients will benefit from a well understood, well defined interface to access

them. XML is commonly used for interfacing with SOA services, though this is not required.

SOA defines how to integrate widely disparate applications for a Web-based environment and uses multiple implementation platforms. Rather than defining an API, SOA defines the interface in terms of protocols and functionality as shown in Figure 1.3 by by Debu Panda. An *endpoint* is the entry point for such a SOA implementation.

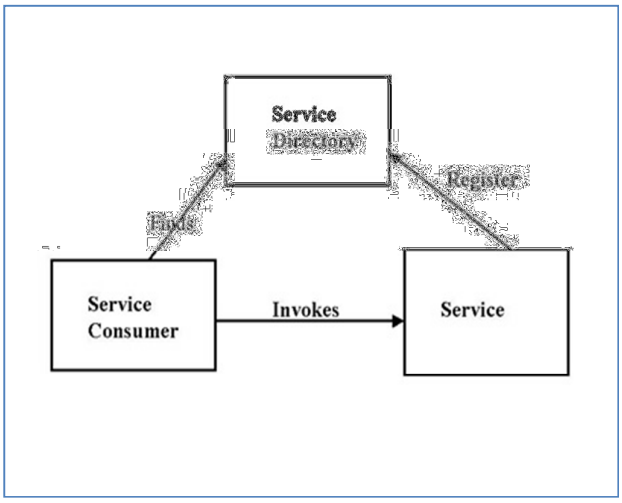

Figure 1.3 SOA from a Java Developers' Perspective

Service-orientation requires loose coupling of services with operating systems, and other technologies that underlie applications. SOA separates functions into distinct units, or services, which developers make accessible over a network in order to allow users to combine and reuse them in the production of applications. These services and their corresponding consumers communicate with each other by passing data in a well-defined, shared format, or by coordinating an activity between two or more services.

Considering all these facts, I have implemented the SOA concept in development as well as deployment. I have implemented the SOA concepts through services that are software services and cloud services. Specifically, for the development and deployment purpose, I have considered the infrastructure

as a service, platform as a service, software as a service and application as a service available for open source tools and technologies.

1.3.3 Benefits of SOA

The benefits of SOA are realized over a period of time. It brings better re-usability of existing systems or applications or functionality and allows for the creation of applications that can be built on top of new and existing applications. SOA enables changes to applications while keeping clients or service consumers isolated from evolutionary changes that happen in the service implementation.

Most importantly, SOA provides enterprises better flexibility in building applications and business processes in an agile manner by leveraging existing application infrastructure to compose new services. The performance evaluation of SOA software systems based on the developed and implemented tool has been discussed in this study in chapter 5.

1.4 Distributed Database

The definition of Distributed Database (DDB) given by Ozsu M Tamer and Patrick Valduriez says that DDB is a collection of multiple, logically interrelated databases (DB)s distributed over a computer network. A DDB management system (D–DBMS) is the software that manages the DDB and provides an access mechanism that makes this distribution transparent to the users as shown in Figure 1.4. Distributed database system (DDBS) is DDB and D–DBMS, together. Collections of data can be distributed across multiple physical locations. A DDB can reside on network servers on the Internet, on corporate intranets or extranets, or on other company networks. The replication and distribution of DBs improves DB performance at end-user worksites.

To ensure that the DDB are up to date and current, there are two processes: replication and duplication. Replication involves using specialized software that looks for changes in the DDB. Once the changes have been identified,

the replication process makes all the DBs look the same. Duplication on the other hand is not as complicated. It basically identifies one DB as a master and then duplicates that DB. The duplication process is normally done at a set time after hours. This is to ensure that each distributed location has the same data. In the duplication process, changes to the master DB only are allowed.

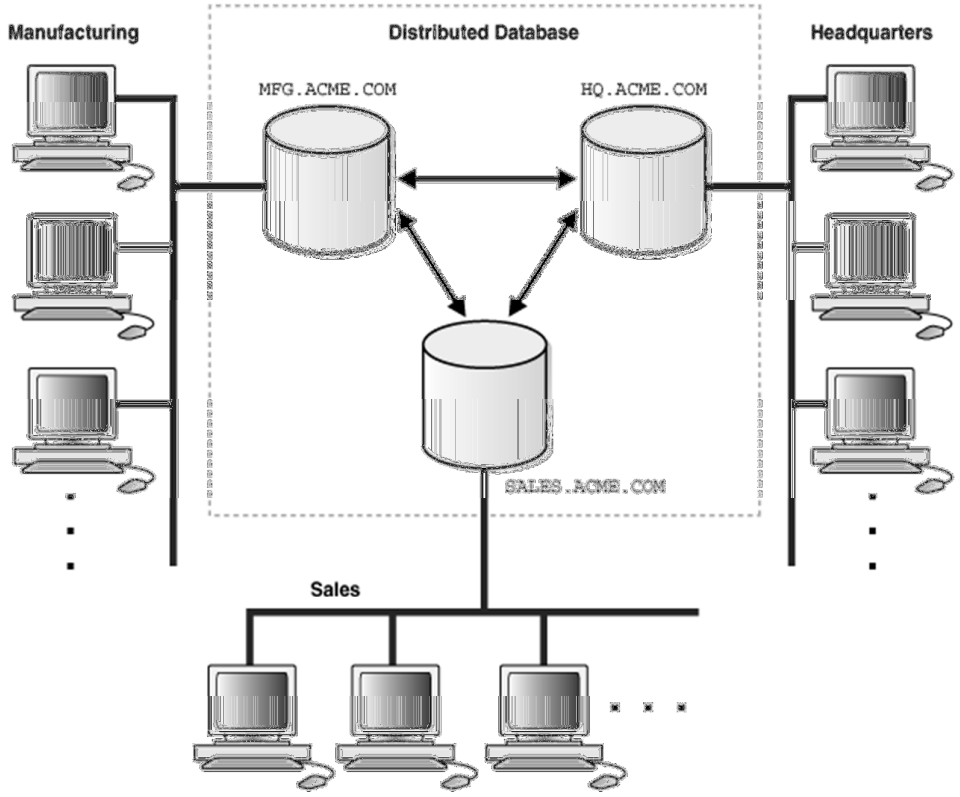

Figure 1.4 Distributed Database.

To store data in DDB, there are two principal approaches i.e. Replication and Fragmentation / Partitioning. In replication, the system maintains several identical replicas of the same relation in different sites so as to realize the benefit of data availability. In Fragmentation, the relation is fragmented into several relations in such a way that the actual relation could be reconstructed from the fragments and then the fragments are scattered to different locations. There are basically two schemes of fragmentation - Horizontal fragmentation and Vertical fragmentation. Considering the various issues for DDB, the DDB administration becomes very important as discussed in 1.5.

Kossman in 2000 has pointed out that, even though good ideas have been presented by research initiatives on distributed database management systems (DDBMS), the prototypes that have been developed did not make it to commercial tools. He believes that such research may have taken place ahead of the ideal time, mainly because of the lack of a communications infrastructure as stable and inexpensive as the Internet. With the growing availability of network communication resources, through the Internet, along with growing quality of service, some of the obstacles in the path of developing distributed databases have vanished.

1.5 Distributed Database Administration

The DB administration is about managing a DB server related to its databases, users, memory, access rights, handling system or power failures etc. The DBA is a person responsible for such administration. He is accountable for the local server, whereas for the DDB administration, there are various DBAs involved so as to coordinate the DDB administration activity. Therefore, the DB concept changes from local to global ie instead of managing a local DB site, one of the DBA or specifically appointed for DDB administration; GDBA is given the responsibility of managing all the global DBs involved in the set up of DDB. The general architecture of D-DBMS given by Ozsu M Tamer and Patrick Valduriez is as given in Figure 1.5.

These DDB systems operate in computer networking environments where component failures are inevitable during normal operation. Failures not only threaten normal operation of the system, but they may also destroy the consistency of the system by direct damage to the storage subsystem. To cope with these failures, the GDBA has to manage the DB based on site autonomy, DDB security etc. The DBA has several choices for tools to use when managing an DDB system. Few tools have been discussed in chapter 3 with respect to their functionality, architecture, dependence etc.

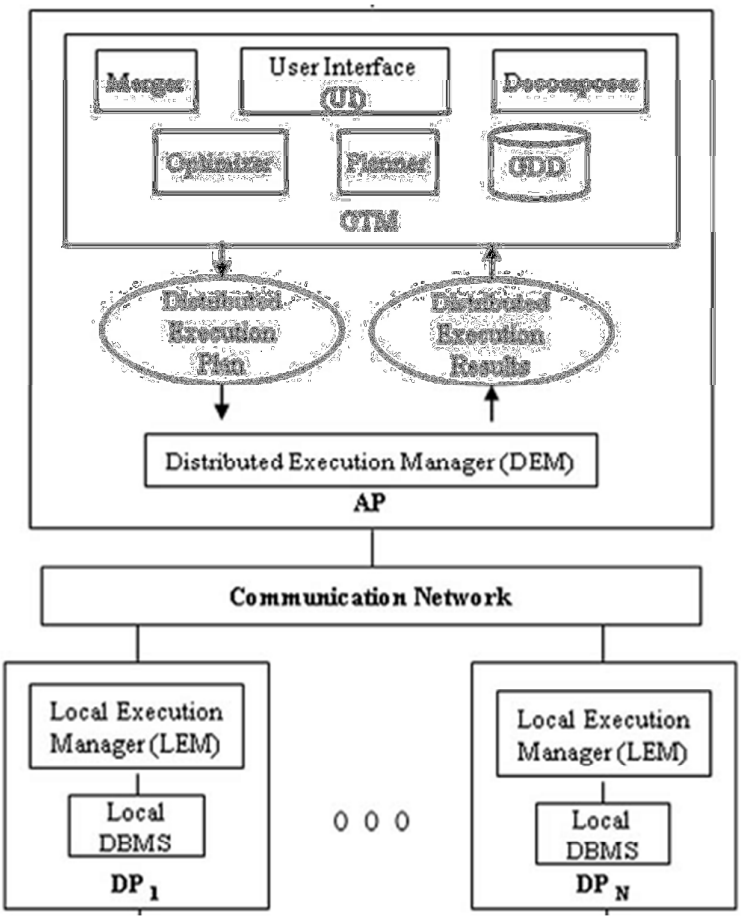

Figure 1.5 Generic D-DBMS architecture

In general, the DDB administration is required for various purposes. The administration for DDB is mostly required for monitoring DB activity in order to know of the performance and health of a DB. It may also involve data migration, operating remote servers (in special cases), job accounting, auditing, communicating with other DBAs in case of any problems related to system or hardware failure, deadlocks, long running queries, job failures etc. All these tasks or processes require some architectural model for developing a software system to aid the DBA with the job. This has been researched by others as discussed next.

According to Breno Mansur Rabelo and Clodoveu Augusto Davis Jr. as the volume of distributed data grows, the use of D-DBMS increases in

importance. One can build an integrated database from isolated and independent segments. The cost of high-capacity servers and the operational decentralization of several types of organizations could be one of the reason. Recent technological elements like XML, SOAP, web services and SOA can be put together, in a review of DDBMS concepts, using SOA to implement Internet based data exchange. As mentioned by the authors, recent works by Campbell in 2005; Tok and Bressan in 2006, have presented initiatives related to the integration of SOA and DBMS, thus proposing a service-oriented database architecture (SODA). From SODA the focus is now further on a new architecture for Internet-based distributed database management using SOA-based communication.

1.6 Layout of Thesis

The businesses are a witness to the ever expanding dynamic global market. Adapting to such a situation is crucial and at the same time difficult. Large amount of money is spent in developing, maintaining and providing security to the systems or applications. The motivation behind this study was to apply SOA principles in the development of a DDB administration oriented tool to be primarily used by the DBA / GDBA and to identify performance parameters and understand the benefits of SOA based system development. This study would enable the IT Managers to understand the merits or demerits in making any decisions related to the tools to administer the DDB that handles the functioning of the businesses or its processes.

This thesis / research basically involves 1) the study of Service Oriented Architecture for software development and Distributed Database Technology for distributed database administrator and 2) the applicability of SOA in the software tool for large data oriented system to evaluate its overall performance. The tool is developed for the purpose of data migration related to heterogeneous DB. Chapter 2 is about the literature study related to SOA and DDB administration. It also mentions the latest web technologies that provide appropriate development functionality.

Chapter 3 is about the DBA, GDBA, their role and responsibilities and the DBA tools used for their purpose. I have discussed the tools with respect to their functionality and features and also got to know that a lot of work has been done in the DBA tool area. There are a variety of tools to choose from; related to the various aspects of administration. I have finally narrowed down on the aspect of data migration and identifying the development of a relevant distinct tool by keeping SOA concepts in view. I have also selected two DBA tools for comparison with respect to their architecture and their functionality.

Chapter 4 contains the information about the heterogeneous data migration tool for DDB, its architecture and its features which will aid the DBA in the administration task related to the data migration in case of DDB. The work done on this tool is very useful for the administrators in situations as discussed in the chapter. It also mentions a case study for dynamic situations where the requirements and expectations change frequently thereby pointing to the fact that the software system can be best developed by an adaptable methodology such as SOA. Also the fact that the large amount of data related to educational sector may be distributed in nature so as to imply the importance of DDB administration in context of the case study. Hence, I have developed the heterogeneous data migration tool for DDB keeping the SOA approach in view. This chapter also contains the user interface of the tool. The tool is based on SOA concepts for its development and deployment. These concepts have helped the identification of overall performance parameters related to SOA software system as discussed in chapter 5. The major factors related to the overall performance are based on the parameters suggested by software engineering and have been discussed with respect to the tool and SOA approach. The sub factors also mention the specific case of SOA system development and deployment with the developed tool in view. I have also attached the tool code at the end for reference.