Chapter 5: Overall Performance Evaluation of SOA Based System

The software systems developed for various user applications have evolved from time to time due to users' expectations and needs for problem solving or data processing. In the process, specific patterns emerged over a period of time in the construction of software systems. The patterns in due course represented various architectures. One of the architecture that became popular was SOA. So now, after the demonstration of the SOA based HDMT, my work concludes with identifying the parameters for evaluating SOA based software system in various contexts so as to benefit others for software system development, for which I have taken reference of SOA based HDMT. Accordingly, on a broader categorization, I have identified the performance parameters for SOA based software systems and mention them in the overall performance of SOA based HDMT from the development and deployment point of view.

The benefits of SOA have already been discussed in chapter 1. But a look at the more detailed technical benefits of SOA given by The Open Group, it is found that SOA has more benefits at the level of service, messages, virtualization and model level implementation. On the other hand, according to TIBCO Software Inc, the benefits of SOA are vast as mentioned in 'Extending the benefits of SOA beyond the Enterprise'. Here they mention that SOA implementation and infrastructure works not only for an organization but also beyond, through partnerships, B2B commerce, collaborations etc. Another important conclusion[1] mentioned by Paul C. Hershey, Shrisha Rao, Charles B. Silio and Akshay Narayan, is that the presence of a network-induced delay in the transactions could hardly be felt in their cloud based application because the setup for SOA application and using cloud services; was

[1] The details of the paper are mentioned in reference [68] of bibliography

deployed in a private gigabit ethernet supported network. Considering the above facts, my effort is directed towards primarily identifying the various parameters on which performance of SOA based system can be analysed and secondly, mentioning various aspects that are distinct for SOA based system. Hence, I have identified and mentioned overall performance of SOA based system from the point of view of software development and deployment of HDMT. Following are the performance parameters that have been identified to consider the various aspects involved in SOA based software systems.

5.1 Development

HDMT has been implemented using open source tools and technologies as mentioned in chapter 4 considering their utility as mentioned in chapter 2. The development parameter of SOA based HDMT can be evaluated in detail as given below. This phase is assumed to begin only after previous processes of analysis and design have been completed. The choice of technology to be used and the service model to be implemented is also assumed to have been decided.

- **Functional Decomposition** – This factor refers to the process by which complex software can be broken down to smaller parts which are better to understand and maintain. With the work on ARTEMIS tool published by Silvana Castano, Valeria De Antonellis and Sabrina De Capitani di Vimercati, I was able to work on the decomposition of HDMT to identify and implement various functionalities to be organized as services. The decomposition could be for an algorithm or for object oriented design. Decomposing HDMT in this context, I have considered as per the work done i.e. the functionality as well as the objects. HDMT functionality has been developed keeping SOA approach in view. This approach along with the basic object oriented concepts made the functional decomposition of overall HDMT functionality, possible. Independent classes related to server, database and table information authentication have been developed that have data members and member methods. These methods have been developed for the

individual database technology or RDBMS and are independent of each other. The functionalities have been decomposed into smaller sub services. These services communicate with each otherl using JSON which provides a data transfer like xml technology. The Figure 5.1 gives an idea as to the various functionalities supported by HDMT tool.

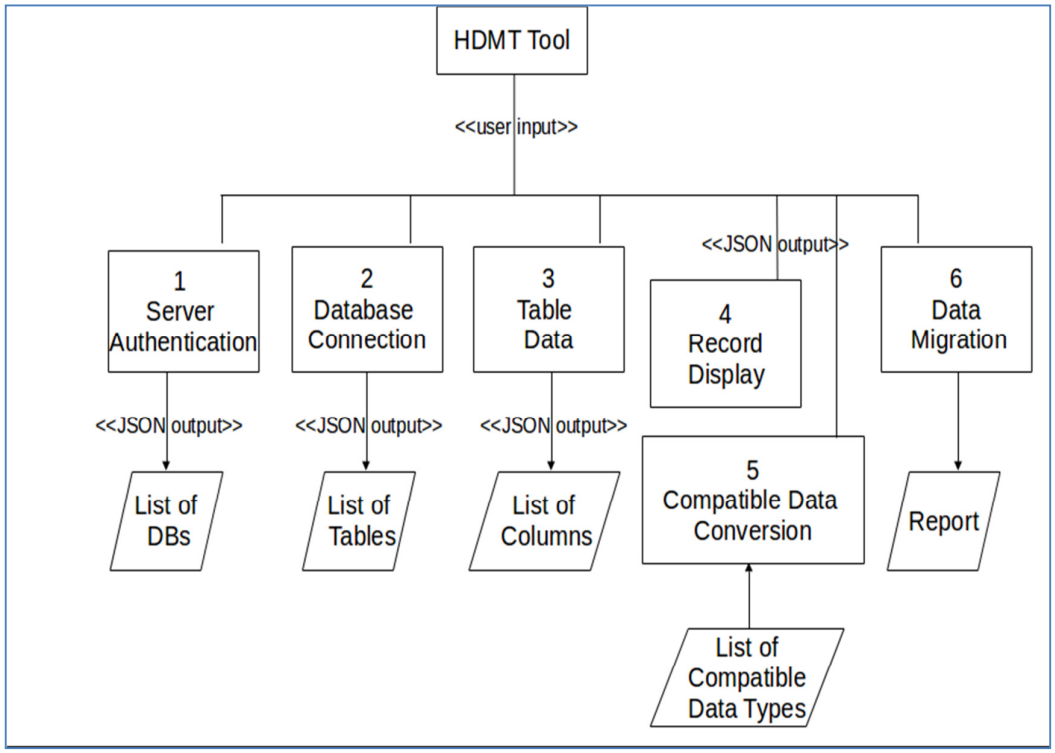

Figure 5.1 HDMT tool and its functional decomposition

The server authentication, database connection and fetching the table meta data are basic services for accessing the user desired data as shown in Figure 5.2. The other service i.e. of displaying the records i.e. data fetched is for the GDBA / DBA to view and then select for the purpose of migration. Before the final migration, the data is checked for compatibility through a service and if possible may be converted to a compatible data type as mentioned by the service. All these activities happen by presenting necessary reports and messages to the DBA / GDBA. The sub services related to MySQL and PostgreSQL are also shown as developed individually for the database technology. The actual migration may be unsuccessful in case of any database violations mentioned on the destination database.

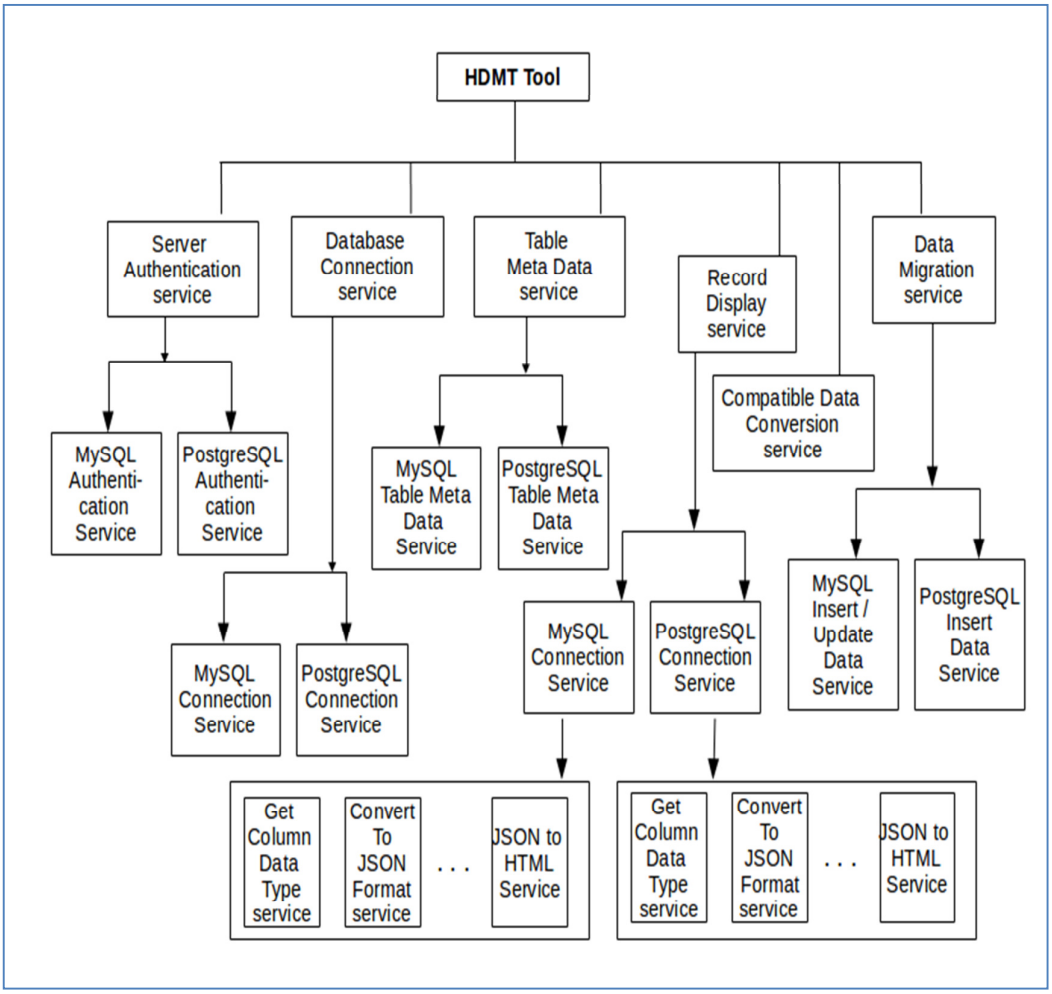

Figure 5.2 HDMT tool and its service decomposition

- **Services and related Functions** – The HDMT functionality has been implemented by developing and implementing different functions as services for their respective purpose. I have implemented HDMT functionality and its services using object oriented concepts too for the server side scripting. Hence, each class developed is independent and is called as a service where the data communication is JSON based. The purpose of the member function is taken into account and not the implementation code. This results in a number of complete member functions which can be reused in other situations. For the client side scripting, I have written small functions for better manageability. During the development phase of the HDMT tool, I realized that the number of

functions is more for SOA based architecture. As the number of functions is more, the lifetime of variables is shorter due to reduced number of lines of code.

- **Variables** – Current web technologies have developed a better data exchange system which makes the variables more manageable. The OOP methodology used in HDMT has made the development more manageable in terms of passing values from client to server side, function to function or within a function. There are many data members on the server side scripting to handle the variable concept. The client side scripting is also developed using small function and its variables. The concept of services was implemented using JSON and jQuery for which the variable definition and maintenance is efficient and manageable and exists for the lifetime of focused services.
- **Inter function communication** – This factor is compared for two situations. First for functions that are communicating and developed using same language; and second for communication between functions that are developed using different languages. In both the cases, the parameter passing for HDMT was done using JSON and jQuery. The same technique was applied for client side or server side scripting.

```
$.post("hdmttableclass.php",{ipaddress:ip, portno:prtno, dt:dbtech, un:username, pwd:password, db:database, tab:table})
```

Figure 5.3 Ajax – jQuery page call using POST method

```
$table_list=json_encode($table_list);

echo $table_list;
```

Figure 5.4 Echo json data to display html table on client side (web browser)

- **Exception handling** –In SOA based HDMT tool, the dependency of one service on another was taken into account to either activate or deactivate a service. As I have developed independent services for individual database technology, the handling of exceptions is easy and customized error messages have been given. The development

language also has error handling functionality as suggested for OOP.

- **Inter application communication** – According to w3c.org, XML is plays an increasingly important role in the exchange of a wide variety of data on the Web and elsewhere. Also JSON is a format similar to XML and is a lightweight data-interchange format. The HDMT handles jQuery or JSON data and makes inter application communication simple and reliable as applicable in SOA.
- **Concurrent development activity** – Parallel development activity is fast as compared to sequential development. The SOA based approach also proves this fact. Most of the services and sub services have been developed independently in my work. As these services communicate using JSON data, the problems related to language and platform dependence have been taken care of. I was able to work on various HDMT sub services at a given time during the development phase. These sub services led to the development of a larger service.
- **Team size** – The team comprises of various job roles and their contribution towards application development could vary depending on SOA based development and deployment. Although the team size during the HDMT development was single, the SOA approach showed that there is a need for variety of job roles to work on wide range of technologies and platforms as needed or as decided during the analysis and design phase. The team size hence becomes a dynamic factor. More members are needed for software development during the initial company steps towards SOA, but gradually as the SOA approach begins to take hold, the team size may be comparatively less.
- **Agility of software development** – This is to know how fast a working function can be developed. For SOA based system development, this factor is most flexible as services comprising of sub services can be developed in lesser time as compared to entire function development. A system comprising of many small well defined and focused services is easy to develop and maintain than a system having limited number of services performing a similar task. In HDMT development this factor was utilized for its advantages.

5.2 Maintenance

Software maintenance is a regular and expected activity for the purpose of meeting the demands in situation, technology, business etc. According to Lehman's software evolution law; that programs usually grow over time to accommodate pressure for change and satisfy an increasing set of requirements, it becomes evident that software comes under maintenance for various reasons that involve not only the change in requirements but also the changing implementation technology.

- **Patch size** – This factor is mostly dependent on the decomposition in development. It could also depend on the technological differences and its up gradation. The SOA based approach allows for software system maintenance by incorporating the necessary patches into its existing services or allowing for the development of new required services. The downtime of the software system in this case is minimized. The HDMT maintenance also saw a few patches which were incorporated as needed within the services.
- **Change Management** – This is an important factor that comes under maintenance as well as well as a topic in itself due to the involvement of the online services. Hence it is explained in section 5.4 along with its sub factors also. It basically works on the concept of assessment, implementation and evaluation. An SOA based software system is made up of well defined services and sub services which are possible for being assessed with respect to the changes and its effect. The implementation may require technological expertise which is available as there is a combination of technical experience in the team. At the same time the SOA manages to incorporate new or existing sub services without changing the meaning of existing services. Such software systems are evaluated as per the functionality without affecting the existing functions or services. The principle of SOA to compose, consume and expose is thus utilized.
- **Concurrent Patching –** This factor helps in faster maintenance of software systems. More concurrent patching is possible in SOA based applications due to smaller size of functions / services. Various team

members adept in different technologies can carry out concurrent patching at the same time. The HDMT patching did not see concurrent patching but a simultaneous patching done on different services.

5.3 Deployment

The concept of deploying the application at client side by installing the executable file has changed drastically to the current scenario of selecting the cloud structure for the deployment of applications. The HDMT has also been deployed on a cloud. The concept of services is not just in producing or consuming the services, but also in availing the services available on the cloud such as IaaS, PaaS, SaaS etc which is the essence of SOA.

- **Portability** – This refers to the usability of software in different environment. Programs written in traditional languages are more environment dependent than those written as services which are more complete. In case of HDMT, the focus was on open source technologies which have their own benefits as discussed in chapter 2. HDMT has a three tier implementational architecture which requires LAMP server, data repository and client to access the system software. In this case only LAMP server requires a compatible platform which can be availed from the PaaS cloud service. The data repository is a php file which is executed by the LAMP server. Hence portability in SOA based HDMT can be done effortlessly and efficiently.
- **Server side** / **client side dependency** – HDMT does not require any third party tool or any other customized software for the server or client side machine. The server needs to have open source LAMP installed for the implementation of HDMT.
- **Hardware dependency** – It refers to the part of an operating system that varies across microprocessor boards and is comprised notably of device drivers and of boot code which performs hardware initialization. HDMT has been implemented on openshift (PaaS) provided by RedHat, on which CentOS Linux 6.5 has been installed. It can be

implemented on any other comparable hardware.

- **Software dependency** – This factor needs to be specified if the software requires a specific browser version, plug-ins, or other software for execution. HDMT requires a web browser for its execution. It has been tested using commonly used web browsers like chrome and Mozilla Firefox.
- **Platform dependency** - It typically refers to applications that run under only one operating system. Sometimes, it means the same as hardware dependent and refers to applications that run in only one hardware series with the operating system not being relevant. HDMT requires Linux operating system for its deployment. The HDMT has been deployed using cloud service.
- **Program file size** – The total program file size for SOA based HDMT is 800KB, which is very easy and efficient for the purpose of deployment.

5.4 Change Management

Change in software system should reflect minimum change effect to user in terms of the working or functionality of the application. The change management process for SOA based systems should be thought in terms of all the services involved therein. According to Bernd Grobauer and co authors there are essential cloud characteristics: on-demand self-service, ubiquitous network access, resource pooling, rapid elasticity, and measured service. All these characteristics point to the fact that below mentioned factors should be considered for change management.

- **Down Time** – The higher the up time and lesser the down time means more availability and reliability of the application which results in better user loyalty and response. Increase in down time of a given service also means that the consumer of the service may switch to some other services in case of SOA based applications. I have mentioned this factor not specific to HDMT, but in general.
- **Risk Factor in change management** – This factor is about a

technology or architecture incorporating the risk factor in change management. Errors and exceptions are expected for change management, but what could be the best mechanism to provide a buffer for such changes. These were the thoughts that I felt were necessary for SOA based systems in general.

- **Simulation** – This factor refers to knowing whether we can virtually test the new SOA based system in general. I realized this factor while changing a few services of HDMT according to their new requirements. The process of simulation proves useful in understanding any exceptions that may cause software system failure.
- **UI Redesign** – Another important aspect of building good software is the user interface which ideally should be simple and easy for the user. In case of any redesign, the SOA approach allows for maximum flexibility and adaptability to future changes. While adapting HDMT to certain changes, I realized that SOA approach is easy for adapting the software according to the new changes.

5.5 Usability

ISO defines usability as "The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use." Usability is the ease of use and learn-ability of a human-made object. The term *user friendly* is often used as a synonym for usable. The term user friendly is customer / client / situation dependent. SOA based software systems can be easily customized as per the preference which gives rise to the following situations.

- **UI Customization** – today's user expects the user interface customization as per individual preference. The incorporation of this facility into existing or new applications could greatly depend on SOA where UI customization services can be implemented without changing the functionality.
- **User Guide** – The use of this document decreases as the UI customization increases. Software systems can be incorporated with

the tool tips for increased assistance. HDMT is implemented specifically for DBA / GDBA who are acquainted with the SQL terms. The tool gives appropriate messages for the completion of data migration. The GUI of the tool is simple and easy to understand for the IT aware people.

- **Response Time** – The better the response time to user actions, better is the user satisfaction towards the application. SOA based applications have their own benefits of decomposition in this regard. The network induced delay expected in services deployed on the cloud is hardly felt as mentioned by Paul C. Hershey. HDMT has been implemented keeping in mind this factor by allowing the mention of number of records to be migrated as a transaction to the destination DB server so as to give immediate response to the user.
- **User Interactions State** – this refers to the maintenance of the data entry state in case of exceptions during forms filling so as to aid the user in continuing his actions when the application resumes working. SOA based software systems can be provided with this functionality as an add-on service.

5.6 Re – usability

The service re-usability principle is a design principle that is applied within the service-orientation design paradigm, in order to create services that have the potential to be reused across the enterprise. These reusable services are designed in a manner so that their solution logic is independent of any particular business process or technology which is one of the core principles of SOA.

- **Functional Dependency** – this factor has differences in the way of data exchange for SOA based applications. The use of XML, JSON etc for the purpose of data exchange between heterogeneous functions in HDMT, greatly solves the problem of functional dependency.
- **Function Re-usability** - this is a very desired feature for most of the situations. The re-usability of the code for SOA based applications is

high. In HDMT, the common functionality has been reused as applicable.

- **Module Re-usability** – HDMT consists of three important processes of fetching the data from the source, transforming the data as applicable and migrating the data to destination side. The number of modules in HDMT is minimum.

There are other types of re – usability like configuration re-usability, documentation / help / manual re-usability and application re-usability. The HDMT facility is at its nascent stage and can be incorporated with these in future.

5.7 Security

This can be broadly classified into 2 factors

- **Application** – The security of an application greatly depends on its platform, architecture (client server or web), type of business, audience etc. Such security threats by various categories of hackers gives the following other detailed factors. The various layers involved at platform, operating system, architecture etc need either physical or software security. Corresponding to these layers, the IaaS and PaaS provide vulnerability tracking and handling at physical level and LAMP provides vulnerability tracking and handling at the software level. HDMT has been deployed using the cloud services so as to provide for these securities.
- **Data** – The security of data depends greatly on three factors. They are data exchange security within an application or inter application, inter device data communication / exchange security and encryption algorithm change management. The SOA based software system has number of small services or functions so code is manageable and security can be implemented within function for data exchange purpose through various encryption algorithms. Also to avoid any data loss, the encryption algorithm change management is very vital which can be handled through the services. HDMT has implemented only basic data security through the use of independent communication technologies.

5.8 Portability

The portability refers to operating system or platform level portability, hardware level and client side portability (applicable for browser, plug ins etc.). HDMT is implemented using LAMP and works with modern browsers. LAMP can be installed on any open source compatible environment.

5.9 Scalability

Scalability is the ability of a system, network, or process to handle a growing amount of work in a capable manner or its ability to be enlarged to accommodate that growth. This aspect indicates whether the software under consideration is scalable depending on the current situation. Scalability is of different types like Horizontal & vertical scaling, database scalability, etc. In my study on cloud services, I worked on this factor for deploying HDMT. I have found that there are options provided for factors mentioned below.

- **Resources** –This factor refers to hardware resources like hard disk and memory. These resources need to be easily scalable. Resources such as Apache, MySQL and hard disk, all have defined memory requirement. Though, the processing by CPU and capacity of RAM may need to be increased in case of increase in number of users. SOA based software systems can be provided this flexibility as per the facility provided by data center and provided the system is adaptable. For the purpose of deploying HDMT, I found this very useful as the scalability facility was provided by data center and HDMT is adaptable to it.
- **Users** – This factor refers to the number of users connected to a website at any given time. This number is dynamic and data centers must provide for scalability as per the increase or decrease in number of users. For the purpose of deploying HDMT, I found this very useful in context of the case study discussed in chapter 4, though HDMT

handles only limited number of users at a given time. Also, the users can be given access rights accordingly and is easy to incorporate in SOA based software systems.

- **Application** – Application scalability refers to the improved performance of running applications on a scaled-up version of the system. I have mentioned this factor for SOA based system in general. SOA based software system is easily scalable.
- **Service Instance Scalability** – This factor is discussed with respect to SOA in general. Scalable services utilize a scalable resource group to contain the application resources and a failover resource group to contain the network resources (shared addresses) on which the scalable service depends. The scalable resource group can be online on multiple nodes, so multiple instances of the service can be running at once. The failover resource group that hosts the shared address is online on only one node at a time. All nodes hosting a scalable service use the same shared address to host the service. Container creates instance as per the requirement.

5.10 Compatibility and Adaptability

The small, independent and well defined services of SOA based software systems together make the software system more adaptable to changes that may arise due to requirement changes or enhancing existing functionality. During the development and deployment of HDMT, I faced this situation and found that changing a few services and sub services or adding new services was manageable. The compatibility of the software systems was also easily managed as I had selected all open source tools and which had defined interfaces.

5.11 Modularity

The modularity of any software system is based on re usability and involves its manageability. The service provided to the user in the form of features is different from the service implementation at the system level. In SOA based software system, a service is more inclusive than component. The functions in

SOA based software systems are focused, easy to manage and do not affect other functions. As the functions are focused, they can be used anywhere. The small size also means processing of a function is fast and the life time of variables is less. All these factors give rise to functional dependence and independence where due to small functions, there is independence. As the functions have general scope they are less dependent and more independent. In HDMT, there are small functions and big functions (that consist of small functions).

5.12 Data Exchange / Interoperability

The case study discussed in chapter 4 mentions that data has to be downloaded or made available in the specified format and then imported or uploaded. HDMT makes data migration possible without any security risk or violation. This means that the data exchange is secure. Moreover, HDMT provides the facility to specify the number of records to be migrated. The default number is 10 which when not mentioned means all the records. HDMT also provides for the data compatibility through equivalent and convertible data types.

5.13 Durability

This factor is new in terms of software durability. Durability is the ability to endure. In terms of software, durability is defined for database systems which is one of the ACID properties. Software being durable is a new concept which refers to its life time ie the time for which a software can be used efficiently and effectively. Many a times software becomes obsolete though it is durable. This happens because occurs because a replacement has become available having more advantages. In case of software durability I have considered the following criteria for SOA based software systems.

- **Adaptability to changing environment** – The first law proposed by Lehman postulates that a program must continually adapt to its environment, otherwise it becomes progressively less useful. The software development environment has already seen a great change

till recent times; SOA approach being a big evolution in bringing about the revolution needed for software systems. Service centric software systems provide maximum adaptability to the changing environments as mentioned by Olivier Nano and Andrea Zisman by defining processes and create methods, tools, and techniques to support cost-effective application development and use. Such systems show better durability.

- **Maintainability** – Maintainability is as per need of :
  - *Changing business logic* – As published by Vanish Talwar, Qinyi Wu, Calton Pu, Wenchang Yan, Gueyoung Jung and Dejan Milojicic, the maintainability is related to the number of lines of code which is one of the reasons of changing business logic. SOA based software systems are made of small sub services to build a larger service. So maintainability required is per sub service to make the software system more durable.
  - *Changing operating environment* – This factor refers to the required adaptations made in the operating environment which may affect the software systems. Lehman's law stipulates that over time software quality appears to be declining, unless proactive measures are taken to adapt the software to its operational environment. An SOA based software system is based on small units called services which are implemented using various languages. These services communicate with each other using other technologies to make them independent. Hence SOA approach is suited for changing operating environment in order to be durable.
- **Replace-ability of software development components** – With changes in business and software development architecture scenario, there are new software development components developed. So this factor is important in making software durable.

5.14 Expandability

Expandability is a factor which may be related to hardware, network, software etc. But basically, it is the ability of a system to accommodate additions to its

capacity or capabilities. From a hardware point of view, expandability may include additional or larger hard disks, more memory, or faster video board. From a software point of view, it may include ability to support more network users, greater number of 'hits' from website visitors. This is different from extensibility which is the ability of a software system (such as a database system) to allow and accept significant extension of its capabilities, without major rewriting of code or changes in its basic architecture. Incorporating SOA Approach does make software system more expandable in the following cases.

- **Enhancing existing functionality** – When there are enhancements to be done on a software system which are not as a result of changing requirements but a result of minimum expectations for software adaptability, then SOA based approach is useful in case of this factor. The enhancements required to be done on existing software systems need to be checked for its effect on the existing software system and uniformity in its applicability too.
- **Incorporating new functionality / modules –** When requirements change, the software must be adaptable to incorporate the changes, but in case of expandability, the addition of new modules is about improving user experience or client expectation. SOA based approach makes such incorporation much easier through the use of services.

5.15 Collaboration

Collaboration is working with others to do a task and to achieve shared goals. It is a recursive process where two or more or organizations work together to realize shared goals. In management terminology, it is also about control and authority sharing. I have focused on the software aspect for the purpose of this discussion. Companies spend to protect data generated and managed at their level. But at times they also need to share as part of business or policy changes. An SOA approach is efficient only under the mentioned terms and conditions of the company. I have identified below mentioned factors to be considered for collaboration.

- **Data Sharing** – This term is new in terms of sharing data as the companies spend to protect data generated and managed at their level. New business or policy changes may require data sharing. Software systems need to be designed keeping in view the adaptability required in such scenarios.

5.16 Resource Utilization

In the software industry, the resource utilization is important as it deals with not only the hardware and software but also the talent involved in the working of the industry. As the demand for quick response increases, the software system and the cloud services should also accommodate for such a demand. The cloud services provide for scalability as per the terms and conditions with the client. As per the work published by Stefan Seltzsam & et al where they mention a problem that the decomposition in finer-grained services allows the usage of hardware clusters and a flexible service-to-server allocation. This factor is discussed for SOA based software systems.

- **Load Balancing** – I have added this factor with respect to SOA based systems. In computing, load balancing distributes workloads across multiple computing resources, such as computers, a computer cluster, network links, central processing units or disk drives. Load balancing aims to optimize resource use, maximize throughput, minimize response time, and avoid overload of any single resource. Using multiple components with load balancing instead of a single component may increase reliability through redundancy. Load balancing usually involves dedicated software or hardware, such as a multilayer switch or a Domain Name System server process. A SOA approach in development and deployment is suited for load balancing as per the service decomposition and cloud services & its terms and conditions.
- **Elasticity (elastic computing)** – In cloud computing, elasticity is defined as the degree to which a system is able to adapt to workload changes by provisioning and de-provisioning resources in an

autonomic manner, such that at each point in time the available resources match the current demand as closely as possible. Elastic computing is a concept in cloud computing in which computing resources can be scaled up and down easily by the cloud service provider. The elasticity of these resources can be in terms of processing power, storage, bandwidth, etc. The SOA based software systems show elasticity in deployment. Alternately another term i.e. Auto scaling (up and down) is also used. It is a cloud computing service feature that automatically adds or removes compute resources depending upon actual usage. Auto-scaling is sometimes referred to as automatic elasticity.

5.17 Integrity

The integrity for SOA based software systems is better. HDMT does not use any third party tools for data migration so as to ensure data integrity. Other SOA based systems can be additionally analyzed on the basis of functional and application integrity.

5.18 Reliability

As mentioned by Jiantao Pa, software reliability is the probability of failure-free software operation for a specified period of time in a specified environment. Software Reliability is also an important factor affecting system reliability. It differs from hardware reliability in that it reflects the design perfection, rather than manufacturing perfection. The high complexity of software is the major contributing factor of software reliability problems. Conversely, the simplicity of a system contributes to lesser software reliability problems. SOA based software systems are simple individually to accomplish the functionality of a complex software system and their reliability depends on following factors.

- **Availability** – A software system deployed using SOA approach is reliable as per the terms and conditions offered by the cloud services. In case of increase in the number of users, the cloud services do offer the scalability option which will make the software system available to

the users. Also the fact that availability is the ratio of up time / total time will depend on the data center. SOA based systems are more manageable because of the decomposition.

- **Up – time –** It is used as a measure of computer operating system reliability or stability, in that this time represents the time a computer can be left unattended without crashing, or needing to be rebooted for administrative or maintenance purposes, though conversely, long uptime may indicate negligence. Paul C. Hershey has mentioned that for SOA based software systems, the usual quality metrics such as uptime and reliability may be considered applicable in the context of cloud systems.

5.19 Supportability

It is easy to provide functional support in case of adaptable software systems. The client side environment detection is also an important aspect of supportability. HDMT is also implemented with jQuery which detects the browser and executes the functions accordingly so as to make HDMT browser independent. Hence if there is any change on client side then web based systems need to detect it and work accordingly which HDMT does.

5.20 Testability

To analyze any software for its effectiveness, it must be testable. SOA based software systems can be tested under various tools and methods to know its purpose realization. HDMT is testable as function size is limited.

5.21 Audit-ability

The conduction of an audit at any level needs expertise and an eye for detail. There are many levels like functional level, module level, code level and application level at which an audit can be conducted. The SOA based software systems are integrated with various development and deployment technologies so as to bring all the services to work in coordination with each other through well defined communication. HDMT is a simple and small utility

service for the DBA / GDBA to consume where the audit is easy. The audit itself can be carried out as internal / external audit by internal or external experts.

5.22 Localization and Internationalization

In computing, internationalization and localization are means of adapting computer software to different languages, regional differences and technical requirements of a target market. Internationalization is the process of designing a software application so that it can potentially be adapted to various languages and regions without engineering changes. Olivier Nano and Andrea Zisman mention that adaptable, flexible, interoperable, and maintainable service centric software systems are made possible through defining processes and creating methods, tools, and techniques to support cost-effective application development and use. Such methodology is the essence of SOA and is applicable.

5.23 Customization

New software services involved in the software customization can be incorporated into the existing software system so as to make the software system adaptable as per the client or customer demands. The services involved in the customization can be implemented at various levels like user level, software level, database level etc.

The HDMT can also be customized at the destination side. This option will cause considerable changes in the number of inputs required. The tool can be customized to take only the source database related information while hiding the destination connection details. This way the destination DB information can be hidden or not shared with the other DBA. Only the relevant tables can be displayed for the mapping purpose for the data communication. The tool can be customized so as to take only the college or institute database connection details to be sent to the governing body website.

In this way, the SOA approach is relevant to users in different ways. For the online services or facilities, the user can ask for customized things through services; whereas for the developers, SOA means better flexibility and manage-ability at the system level.