POLITECNICO DI MILANO

Facoltà di Ingegneria dell'Informazione

Corso di Laurea Specialistica in Ingegneria Informatica

VALUE ANALYSIS OF WEB SERVICE REPAIRABILITY

Supervisor: Prof. Pernici Barbara

Master thesis of:

Farhana Islam	Matr. n. 722771
Meherun Nesa Lucky	Matr. n. 709400

Anno Accademico 2008/2009

ABSTRACT

The thesis is the business analysis of web service repairing system. In a web based application, there can be failures of different web services. In this work, we have tried to apply different repair actions on unsuccessful web services and to find out what is the impact of those repair actions from the business perspective. For that, we have proposed an e-business scenario which is web service enabled and we have taken a web portal for analysis which is responsible to deliver books from different publishers to the customers according to their requests. We have discovered the faults that may occur in the processes. Then we have proposed some repair strategies which can be applied depending on the failures. After applying repair actions, we have computed the business values from the perspective of the web portal. Lastly we have made a comparative analysis on the outputs of different repair strategies. We also have implemented a simulator which gives the results of this business analysis depending on the given inputs.

SOMMARIO

La tesi è l'analisi di business del sistema di riparabilità servizio web. In una applicazione web based, ci possono essere diversi errori di servizi web. In questo lavoro abbiamo applicato interventi di riparazione su diversi servizi web riuscita e cercato di capire qual è l'impatto di tali azioni di riparazione dal punto di vista business. Per fare questo, abbiamo proposto uno scenario di e-business chiamato "Online Book Shop (OBS)" che è il web service attivato e abbiamo preso un portale web per l'analisi che ha il compito di consegnare i libri da diversi editori per i clienti in base alla loro richieste. Abbiamo scoperto alcune anomalie che possono verificarsi nei processi. Nel nostro lavoro abbiamo considerato difetti in alcuni processi producono guasti nei servizi web. Errori nei servizi web possono essere di diversi tipi. Può essere scarsa qualità delle informazioni, informazioni errate, out-of-date informazioni, ecc. Anche prendendo troppo tempo nel dare un servizio o addirittura l'incapacità di dare un servizio è anche considerata come fallimento di web service. Ogni volta che questi tipi di errori si verificano nei servizi web, i servizi devono essere auto-riparazione. Per fare un sistema auto-sanabile, è necessario inserire alcune componenti che verranno diagnosticare le cause e si possono applicare le azioni di riparazione appropriato per guarire da questo fallimento. A questo proposito si deve essere analizzata come questa scelta della strategia di riparazione influenza la qualità dei risultati di tutto il sistema, perché è legato al costo e beneficio di un'organizzazione. Nel nostro scenario proposto di e-business, quando un editore non riesce a consegnare tutti i libri richiesti al OBS, due azioni di riparazione può essere applicata: riprovare e di sostituzione. Ciascuna di queste azioni di riparazione è applicato separatamente per analizzare i risultati economici dell'organizzazione. Per riprovare, OBS ripete l'intero processo con la richiesta stesso libro. Per la sostituzione, OBS chiede un numero di editori per le informazioni di disponibilità del libro richiesto dal cliente e sostituire l'editore già riuscita con quella che può fornire il libro desiderato. Abbiamo esaminato l'impatto di ciascuna di queste azioni di riparazione sul valore economico di un'organizzazione come OBS. Il valore economico è il profitto economico reale che l'organizzazione ottiene dopo calcolare di tutti i suoi costi e benefici. Abbiamo considerato come un indice di soddisfazione importante proprietà di entità commerciali associati, che ha un enorme impatto sul valore economico di OBS. Quindi, abbiamo calcolato le spese, le entrate, indice di soddisfazione e il valore economico di OBS in tre casi diversi: nel caso peggiore, lo scenario

dopo l'applicazione di riprovare e lo scenario dopo l'applicazione di sostituzione. Qua, nel caso peggiore è considerato come un scenario in cui il numero di richieste soccombente è alta. Abbiamo fatto un'analisi comparativa sulle uscite delle strategie di riparazione diversi. Abbiamo mostrato come i cambiamenti nei fattori diversi possono dare un effetto sul valore economico di una organizzazione nella nostra analisi. Ad esempio, le variazioni nel numero di riprovare o in numero di richieste di sostituzione avere un impatto sul valore economico di OBS. Abbiamo anche realizzato un simulatore che fornisce i risultati di questa analisi di business a seconda degli ingressi dato.

INDEX

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

At present time, web services are one of the most talked about research area in the field of Information and Communication Technologies. Web services are the reusable components of web application which can be used to exchange and share data between different applications and from different environments. The main idea behind this concept is to solve the business problems by distributing the solution of discrete problems to some specialized components that are responsible to give some solutions to these business problems. The advantage of this decentralization of the processing mechanism is that there will be fewer places that need to be updated when things will change. The concept of web services came early in 1990s with Sun Microsystems's "the network of the computer" campaign.

On the other hand, a significant requisite for web services is the ability to handle resource variability, ever-changing user needs and system faults. Designs that enable software systems to heal themselves of system faults and to survive malicious attacks would radically improve the reliability and consistency of technology in the field. The endeavor to secure these benefits has originated the concept of self repairing systems. Self-repairing can be defined as the property that enables a system to perceive the ability to make the necessary adjustments to restore it to work properly.

Chapter 1 - Introduction

In this thesis, we have denoted our attention to analyze the effects of applying different self reparable actions to the web services from the business perspective.

In this chapter, we give a review of web services that includes key components, advantages and disadvantages of web services. Then we also discuss Service Oriented Architecture (SOA) followed by Self-healing and adaptability of web services.

1.1 A review of web services

According to W3C, a web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards [1]. This is a client-server architecture in which client sends an XML document formatted in a special way in accordance with the rules of SOAP specification [2]. The main difference between web applications and web services is that web applications give HTML responses to HTTP requests; on the other hand, web services produce SOAP responses to HTTP/SOAP requests [3].

The main concerned issue about the web services is interoperability. In this architecture, a SOAP message is sent to a web service enabled web site (for example, weather forecasting website) with the parameters needed for a search. The site then returns an XML-formatted document with the resulting data (temperature, location, time etc.). Because the data is returned in a standardized machine-parse able format, it can be easily accessible from any platform. SOAP provides a way to communicate between applications running on different operating systems, with different technologies and programming languages.

1.1.1 Key components of web services

The interactions of web services take place between some components. Although there are different vendors who are involved in creating different components, there are some high quality standards of web services components which are followed by every programmers and vendors. They are:

- Simple Object Access Protocol (SOAP)
- Extensible Markup Language (XML)
- Web Services Description Languages (WSDL)
- Universal Description, Discovery and Integration (UDDI)

These components must be compatible with each other i.e. they must behave in such a way that other components can accept them. The core web services standards are described in the following sections.

1.1.1.1 Simple Object Access Protocol (SOAP)

SOAP is a specification which contains XML grammar for both sending and receiving messages. Using with HTTP, SOAP messages are sent to the web servers. Web server invokes the specified function defined by the web service, according to the request in SOAP message from the client machine. SOAP provides a message format in client server architecture in such a way that it is not bound to any hardware and software architecture. The main goal of SOAP is to transfer information from one machine to another independent of language and environment.

There is a specific format of SOAP messages. It has four elements as described below:

Envelope: This element identifies the XML document as a SOAP message

Header: This is an optional element that contains header information (for example, authentication)

Body: This element contains the actual call and response information. This is the XML document that has to be moved from one machine to another.

Fault: It is also an optional element which containing errors and status information.

1.1.1.2 Extensible Markup Language (XML)

XML is used to create grammars that are described in XML schemas. XML schemas provide the tags that are allowed to use in the XML and the relationships between the elements defined by these tags. The main advantage of XML is that it is not predefined. Users can define their own tags. So, XML is written in natural human readable language which can be accessed from any platform.

1.1.1.3 Web Services Description Languages (WSDL)

Web Services Description Languages (WSDL) describes how to invoke a web service and what output is gained from that invoked web service. This is the specification through which web services are called. It describes a simple method call to a web service. These methods are independent of the programming language in which the actual web service is written in or the platform in which the web service runs on.

1.1.1.4 Universal Description, Discovery and Integration (UDDI)

UDDI is a mechanism for publishing and describing the web services to the potential users so that customers can get the knowledge about the capabilities of the web service and also can get the information that is needed to make the initial contact with the site. The web services are registered in UDDI by following some rules and giving some specific information about the services. Depending on the access, UDDI can be public, private or semi private. Public registry is open for all. Private registry is active only within a specific organization and it exists behind the firewall of the organization. Only the people of the specific organization can access the UDDI registry of that organization. Semi private UDDI is accessible by some limited outsider people according to the choice of organization (for example. trading partners of an organization).

In Fig 1.1, web services in the development and runtime environments are shown [3]. An application developer finds and locates the web service from the UDDI registry to use that service in developing the client application. The developer gets the details of the service by following the link to the WSDL document with this service. On the other side, the developer does the appropriate coding for the client application that uses the web service using SOAP and HTTP. In Fig 1.2, the deployment of web services has been shown [3]. A web service developer develops a web service and deploys it to an appropriate host. Then a WSDL document is created which describes the detail information about the web service i.e. how to use that service etc. A UDDI document is also generated which is used to publish the web service on a UDDI registry.

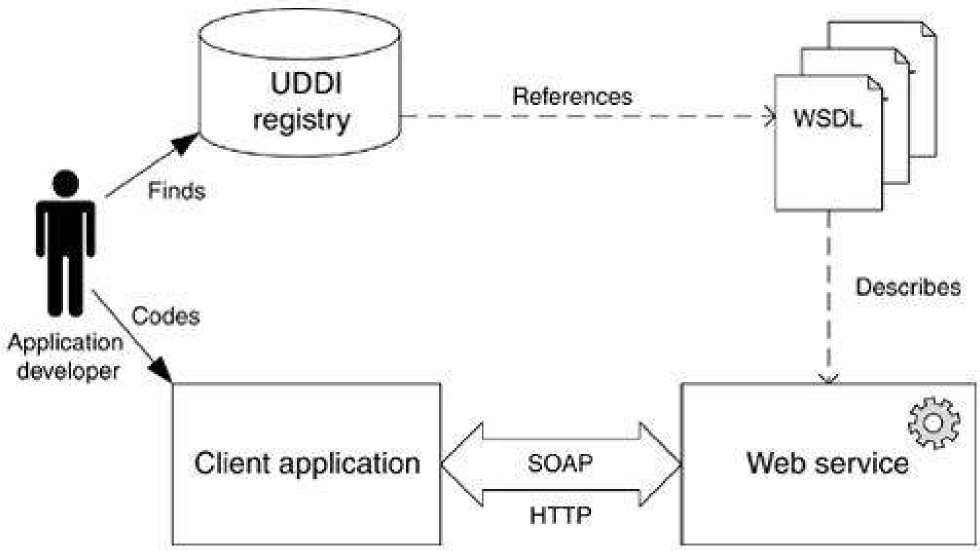


Fig 1.1: Web services in the development and runtime environment (source: [3])

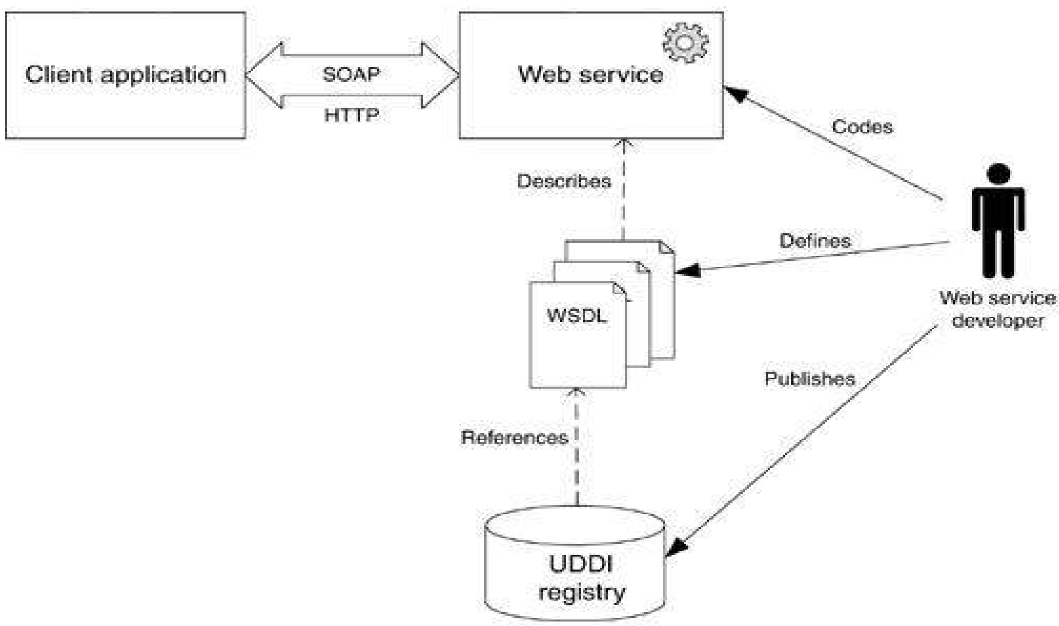


Fig 1.2: Development and deployment of web services (source: [3])

1.1.2 Advantages of web services

Web services have made a tremendous impact on the field of information and technologies. They are increasingly used in solving today's problems. Different types of problems from complex business logic to normal weather forecasting; from online shopping to engineering analysis have now solutions by integrating web applications with the appropriate web services. Without using web services, these problems were solved in a huge amount of time and cost and most importantly, the solutions provided by the web applications could not be able to give the proper result in changing user requirements and environment. There are uncountable advantages of web services. Some of them are described below:

Integrating with legacy systems: Many older solutions that did not use web services are often fragile and even expensive to maintain. Web services can be integrated with the legacy code in such a way that this integration requires almost no change. Developers now prefer to use web services to make their applications cost effective, reliable and maintainable.

Better communication: Web services provide opportunity to save cost in communication. Internet allows machines to connect each other using each computer's service provider. Using web services, the organizations can give the status of their internal work to their customers. For example, users can get the information of the location of their percales coming to their home from the web site of a courier service.

Cost effective: The reusable component of code which can be used by different application is very useful if the component is compatible with the platform of the application. Web services are the reusable components which can be used form different applications regardless of which type of language the application is written or which type of operating system it is running on. It saves cost as it is created once but can be used many times.

Faster development: By using the reusable web services, the development of web applications have become faster lowering the cost.

1.1.3 Limitations of web services

Though web services provide many advantages, they have some limitations. Some limitations of web services are discussed here.

Availability and efficiency: Sometimes web services are not available due to infrastructural problem; for example, the ISP may be unavailable. Even web services might give wrong information due to some internal error in the application and databases. The rollback of the entire transaction is difficult in case of failure of a web service.

Changing requirements: Web services follow the rule "one size fits many customers". The structure of a specific web service cannot be changed to satisfy the requirement of a small number of customers.

Performance: Web services rely on HTTP. Though HTTP can handle many requests quickly, it has to make a connection with the client each time whenever a request comes from client and it has to terminate the connection when the request has been processed. So a lot of time is wasted in creating and terminating the connection with the client. Other technologies (for example, RMI, CORBA, DCOM) do not have this type of limitation as they maintain the connection with the client throughout the entire lifecycle of the application. Also, HTTP is not a reliable protocol that ensures delivery and response. Another limitation regarding performance is parsing to or from an XML file is time consuming especially if the content of the XML file is complex.

Billing problems: The specification of web services provides the way to find services from UDDI registry but it does not provide any mechanism to handle the pricing of the use of the service. The web services that are published in the registry can be used free of cost. There is no mechanism of computing the billing of the use of web services. The specification should provide some billing rates and contract of the web services.

The technology of web services is relatively new in this field. Though this technology is built on some standard specification, each vendor is using it in his own way in developing the application. Also the people are now getting used to use the benefits of web services day by day. As the specification is evolving rapidly, hopefully the standard specification of web services will emerge within a few years eliminating the lacking in it.

1.2 Service Oriented Architecture (SOA)

Service-Oriented Architecture (SOA) provides a framework for the infrastructure to facilitate the interactions and communications between services. It provides a message-based interaction between software agents, each accessible through standard interfaces and messaging protocols [5]. These agents can be service providers or service requesters (clients) interacting with service discovery agencies. SOA exhibits the following four properties:

Modularity: The services in the architecture are developed as independent modules of functionality, offering well defined interfaces to their users.

Loose Coupling: SOA is not tightly coupled with the requester's process.

Technology neutrality: As we already know that web services are technology neutral, they are usable by any requester. Communication between services is message based, with the message format being standards-based and platform-neutral.

Location transparency: The services have formal documented interfaces and they are easily locatable and accessible over a network.

In SOA all the services can be pulled together to form a complex business process. SOA has emerged not just in systems development but also in business processes. It affects the dynamic capabilities of organization [5]. By adopting SOA, organizations can get a competitive advantage in rapidly changing market environments by concentrating on their dynamic capabilities – i.e., product flexibility and agility in organizational transformation in response to rapidly changing market conditions and customer requirements.

1.3 Self-healing and adaptability

The dynamic nature of the business world shows the continuous pressure to reduce expenses, increase revenues, and remain competitive. This calls for a rapid reaction to the market trends, a rapid handling of user needs, and a rapid understanding of forthcoming challenges. To support businesses in the process of reaching these goals, Web services need to be self-aware of the environment in which they operate and self-healing so that they can recover to normal levels of operation after disturbances. Self-healing is intuitively defined as "A system is self-healing if, and only if, after the occurrence of any basic fault, a diagnosis is issued that automatically raises a repair plan fitted to the fault."[6].This definition implies two properties of the system: diagnosability and repairability. The association between a fault and a set of possible observables is defined as diagnosability. Repairability of a fault is possible if and only if there exists a repair plan that repairs it.

Chapter 1 - Introduction

Self healing of web services is a process by which web services can be able to react to the failures, to diagnose the cause of the failure(s) of services which we consider as faults by monitoring themselves and can be able to recover from that failure by taking some repair actions and strategies. Faults are the causes that generate the failure in web services. In other words, faults in some processes produce failures in the web services. Failures in web services can be of different types. It can be poor quality of information, wrong information, out-of-date information etc. Also taking too long time in giving a service or even the inability of giving a service is also considered as failure of web service. So in order to make a system self-healable, it is needed to insert some components which will diagnose the causes of failures and can apply the appropriate repair actions to get healed from that failure. Here repair actions are some processes which should be applied in the system to get recovered from the failures. In this regard it must be analyzed how this selection of repair strategy influences the quality of outcome of the whole system because it is related to the cost and benefit of an organization. It is very obvious that there can be failure of any web service due to some errors in executing any process of that service. A process is any instance of the application which runs to produce some results for a service. There can be several processes running concurrently to execute one service. Indeed, there can be several web services in a web based application. So there must be some repair strategies which should be applied when any failure occurs.

The quality of a web based application largely depends on the quality of web services that the users are getting. This quality is measured by efficiency, performance and reliability. So the web services in an application must be built in such a way that they can ensure the quality of the web application. Any failure of a service will affect not only the application but also the partners who are related to the system. For example, in a web based application, if a customer has to wait for getting a service for a long period of time, he will be unsatisfied and maybe he will not be the customer of this application any longer. As a result, it will be a loss for the business partners as well as for all the shareholders who are connected to this application. On the other hand, if the customer is a registered regular customer, it will be his loss also as he is getting bad quality of service from this website. Customers are very important for an organization as they are not only the source of income for the organization but are important source of publicity also. If the customers are dissatisfied, then it will make a bad affect on the publicity of that website. So the quality of web services is a major element which must be taken into account in building a web based application.

A web service is self-adaptive if it has the ability of being aware of the environment in which it is running and reacting to the possible changes in the environment which is able to modify web service execution [7]. Such changes can be derived from both internal and external reasons. Internal reasons can be the failure of a service or a part of it or if a service quality level is lower than the related provider promised. Whereas users are responsible for external reasons. The concept of web service is renamed as e-Service as it can be operated in different channels like web, PDA, smart phones etc [7]. Adaptation of this MAIS model can be achieved by channel adaptation, service adaptation and process adaptation. For channel adaptation, MAIS platform can decide to select a new channel when the quality level is lower than the quality promised whereas substitution mechanism is chosen to perform service adaptation. For process adaptation, re-optimization takes place when the quality parameters deviate or will probably deviate. During re-optimization both functional and quality aspects are considered.

1.4 Objectives and Goals

In a service oriented architecture, the economy of the system depends on many elements. In our thesis, the economy of a web based application has been analyzed in terms of economic values. Economic value or simply stated here as value is the true economic profit that an organization gets after computing all its cost and benefit. Here cost and benefit includes many factors that are responsible for running the system. Cost is related to the production, maintenance, marketing of the system and the services that are offered to the customers where revenue is related to the income that is obtained by the system. Contribution of the properties of business entities also makes a huge affect on the economy. The term “business entities” means the partners who are associated with the application. The properties which can be related to the business entities are relationship level, risk level, satisfaction index etc. These properties are very important to get a profitable economy. In our proposed web based application, we are concentrating only on the satisfaction index which is actually dependent on availability, risk level and relationship level. Satisfaction index will be discussed elaborately in the following chapters.

Chapter 1 - Introduction

In building a web based application in the e-commerce based approach, how can it be made better in such a way that this application will be more reliable and will give more benefit to both the customer and the provider so that all of the actors who are related to this application will be satisfied is the main consideration in choosing this topic as our thesis work.

In the e-commerce scenario which we have proposed in our thesis, we have tried to find out what are the general faults that may occur in the system and generate failures. Then we have proposed some repair strategies which we have applied to repair the faulty services. After that we have analyzed the economic value of the application by computing cost, revenue and satisfaction index. The main goal of our thesis is to analyze and compare different changes in the economic value of the application by applying different repair strategies. For this, we have implemented a tool which will show a comparative analysis of the value of the application based on user's given input and in a given environment.

1.5 Organization and Content

The thesis contains six chapters including Chapter 1. In Chapter 2, there is some literature review in which we have discussed some work related to the self repairability of web services. Here we discuss some models and architectures according to which the repair strategies can be applied in the system. In Chapter 3, we propose an e-business model which is a web based application. Our main analysis is based on this application. Here, we describe the properties of entities that we are considering and the general faults and failures that we find in this system at run time. Then we introduce the repair strategies which we apply and analyze the affect of each repair action in the system. Chapter 4 represents the computation of the economic values of the system. There is a comparative analysis in computing the value in different contexts after applying different repair actions. Here we also introduce our simulator which is used for our analysis. Chapter 5 represents the results of comparative analysis between the repair actions based on a given environment obtained by the simulator. Here we give some examples based on which we can get some significant results. Chapter 6 is the final chapter which gives the conclusion. In this chapter we make a summary on our work and we mention some extensions of our thesis for the future work.

Chapter 1 - Introduction

We think that our thesis will provide an innovative contribution to evaluating the business impact of applying different repair strategies in the web services.

CHAPTER 2

Literature review

Self repairability is an essential step for the web services to become self-healable. In recent years, a number of works have been done regarding the self repairability of web services. During our thesis, we went through those relevant major works in order to understand the state of the art of our project domain. In the following sections we discuss some of the works which we studied, and which we found closely related to our thesis.

The three research papers that we discuss in this chapter are:

1. *Quality-aware design of repairable processes*
2. *WS-DIAMOND: an approach to Web Services – DIAgnosability, MONitoring and Diagnosis*.
3. *Estimating value in service systems: A case study of a repair service system*

2.1 Quality-aware design of repairable processes

The research paper [8] mainly discusses about different repair strategies and proposes a method to select the suitable repair strategy in the process run-time or design-time. The result of this research is to evaluate the effects of repair strategies on the quality dimension of the application.

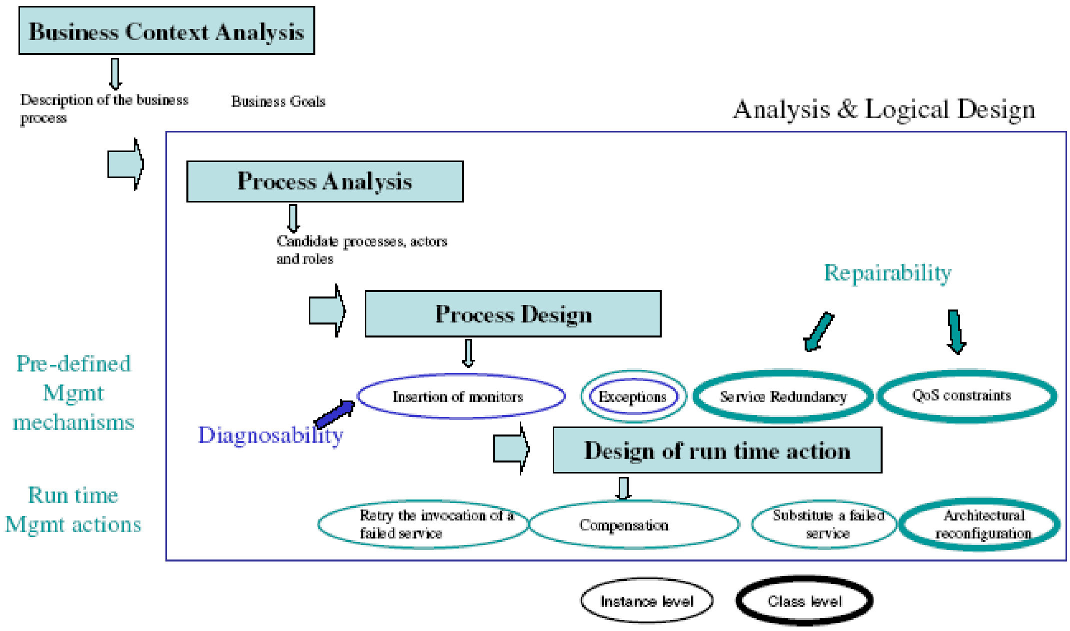


Fig 2.1: Analysis and design method overview (source [8])

2.1.1 Analysis and design method overview

In this paper, a framework is introduced for the analysis and design of a process on which the diagnosis can be performed and repair actions are applied. Fig 2.1 represents an overview of the design of self reparable web services methodology. In business context analysis, the internal and external environment in which the business operates are analyzed based on business goals. In process analysis, processes of the business are analyzed (i.e., the actors, activities, roles and requirements are elaborately defined). In designing of process, some components are inserted for diagnosis of failures. They improve both fault diagnosability and repairability. The quality of data is also checked in process design to produce defect-free information. Since it is better to apply repair action at run time instead of process design time, the repair action at run time is considered as run time action. In designing of run time action, different repair actions are taken into account and based on the application they are chosen to be applied at run time.

The difference between repair strategy to adopt at design time and run time is that in design time, the designer predicts some exceptional situations and provides some repair strategies of the system so that the system can react only on those exceptional situations. But this type of repair strategies cannot be applied to all exceptional situations or faults. On the other hand, repair strategy at run time is not limited within some predefined exceptional situations but it can be applied to any fault bringing the system into the normal mode.

2.1.2 Detection and correction of errors in web services

According to the model described in this paper, the reasons of an error can be the following:

- An error generated by the activities that precede the analyzed one
- And, the error generated by the analyzed one. This type of error considered as self-generated error.

For detection and correction of these types of errors, there are different methods. For example:

- Data cleaning by manual identification: comparison between value stored in the database and value in the real world
- Data bashing (or multiple source identification): comparison of the values stored in different databases.
- Cleaning using data edits: automatic procedures that verify that data representations satisfy specific requirements.

In managing the exceptions in process design, there are some handlers in this model which are associated with a single action or a set of actions. The handlers are:

- Fault handler: Undo the partial or unsuccessful work.
- Compensation handler: Compensate activities which generate errors
- Event handler: This handler deals with incoming messages and temporal alarms.

The main purpose of these handlers is to diagnose the reason for the errors and to correct these errors. The drawback of these handlers is lack of flexibility.

For recovery actions, five patterns are described in this paper. They are:

External variable settings: In which the designer is allowed to identify which variables can be set from the incoming message, and associate it with event handler for modifying the corresponding variable.

Timeout: A set of activities will be performed after a specific time threshold.

Redo pattern: It allows redoing a task without hampering the current ongoing task.

Future alternative behaviors: Some alternative behaviors will be available along the execution of the process. It is assumed here that default and alternative behaviors are mutually exclusive.

Rollback and conditionally re execution of the flow: It is a pattern which allows the roll back of the process until a safe point and then executes the same or a different behavior.

Service redundancy is used in the process design in order to reduce the probability of failures and to assure the correctness of the process execution.

To ensure the quality of service, some constraints are also included in the process design of the model. These constraints must be satisfied during the service execution. If the constraints are not satisfied then some mechanisms are adopted to ensure the quality of web services.

2.1.3 Run time repair actions

Repair strategy is classified based on the type of repair. They are functional and non-functional repair. Functional repair can be applied to all exceptional situations where the workflow activities are corrupted and the activities produce wrong and abnormal results. The non-functional repair strategies are applied to all the faults related to QoS, for example, violation of constraints in the parameters and requirements.

In the model described in this paper, some run time actions are discussed which can be applied at the time whenever the fault occurs. Some of these are as follows:

Redo: Re-execute the service with new input parameters.

Retry of a failed service: The faults at which this type of repair action can be applied is unavailability of one or more services. In retry the invocation of the service will be started again without changing any parameters which is different in case of redo.

Compensation: This is the most complicated and difficult repair action which can be applied only to some specific activities. It allows rolling back all the side effects due to execution of some activities so that the state of objects will be same as it was before executing that specific activity.

Substitute: When one or more services are completely unavailable, this type of repair action allows substituting the failed service provider with another one who has the same kind of operation or service.

2.1.4 Process quality dimension

In process analysis, different quality criteria have been taken into account to find out the quality of a single activity or the entire service. The criteria are: data quality dimension, process quality dimension and institutional quality dimension. The faults that are associated with data quality dimension are typos, different format and conflicts in data quality value. The fields that are considered in process quality dimension are availability of a service, execution cost of an activity and failure risk associated with an activity to be failed during execution. Institutional quality dimension includes reputation of the service provider based on trustworthiness and fidelity based on the relationship between the design alternative provider and the process owner. Each criterion has been assigned a weight to express the importance of that specific criterion among all. Users can express their preferences of importance trough an interface. Also analytical hierarchy process approach has been applied that assigns each dimension a score which represents the relevance of that dimension with other parameters. At the end of process analysis, the score of each quality dimension is calculated to find out which quality dimension should be considered to be applied when a fault occurs depending on user's preference and analytical hierarchical process approach.

In the paper, repair actions have been analyzed by considering the influence of each repair strategy on the quality dimension of the project "FoodShop Company". "FoodShop Company" is an e-commerce scenario. It sells food products on the web. Based on influences of the repair actions, different methods (i.e. AHP) can be adopted to assign the weight to the repair strategies or quality dimensions so that the values of impacts of different repair strategies on different quality dimensions can be found. It will help the provider to apply the most suitable repair strategy in case of a failure.

2.1.5 Relevancy with our work

In this paper, the authors mentioned different repair actions of web services which can be applied in case of web service failures. But the paper didn't discuss about the financial effects of applying these repair actions. In our thesis, we have actually focused on the financial effects of applying different repair strategy of web services from the business perspective considering all the repair actions are applied at run time. We have analyzed an e-business scenario and computed the economical value of the business for each repair action.

2.2 WS-DIAMOND: an approach to Web Services – DIAgnosability, MONitoring and Diagnosis

In the second paper of our literature work [9], a framework for self-healing of web services has been proposed in which the functional and non-functional errors are monitored, detected and diagnosed. Also a tool or method has been defined to design the web services in such a way that it can ensure the diagnosability and repairability of web services. For the diagnosis process, the tests that have been considered here are:

- Alarms raised by services during their execution
- Data exchanged by services
- Internal data to a service

Depending on these observations, the architecture provides a set of recovery and repair actions. The architecture includes a monitoring service which identifies the QoS problems and a repair module which provides the recovery plan based on the diagnosis information.

In the "FoodShop Company" scenario, each process can invoke both operations of internal services as well as external services where internal services perform business process logic and external services are responsible for sending and receiving messages. Each process is identifies by input, output and fault message. A failure which is the observed symptom of a fault can occur during the execution of a process. According to the WS-Diamond, when a failure occurs, a fault is diagnosed and then a repair is applied based on some repair plans which is generated online or prepared offline for a given process. A fault is identified by the faulty service which generates the fault and the erroneous messages originating the failures. For each failure-fault pair, there is a plan which contains some repair actions that can be applied based on the condition of the variables evaluated during repair plan execution. In this paper, the repair actions that have been taken into account are:

- Retry an operation
- Redo an operation
- Compensate
- Substitute a service

There can be two types of faults: temporary faults (e.g. wrong item code) and permanent faults (e.g. service doesn't answer within a given time period). Repair actions are applied based on the types of the faults. For example: retry and redo are applied when temporary faults occur whereas compensate and substitute applied at permanent faults.

2.2.1 Relevancy with our work

From this paper, we have got the idea about different faults which can occur in a web based application and generate failures of web services. In our proposed e-business scenario, we have analyzed the faults and failures for which repair actions are applied. The e-business scenario which we have proposed in our work is called "Online Bookshop" where people can buy their desired books from different publishers. Here we have considered two types of failures. They are: wrong information due to database error and wrong result of customer request. For these failures retry and substitution are applied as repair actions.

2.3 Estimating value in Service System

In the paper [10], the analysis of the complexity of service systems has been proposed. This analysis has presented the information which can be used to analyze service systems in terms of economic values. This analysis has been applied to an example of a car manufacturer and its service system for suppliers and dealers to improve customer satisfaction and provide options and analytical models for outsourcing decision makers.

2.3.1 E-Business model

It offers a flow model for offerings and revenues, with economic entities as nodes. Any kind of business interaction can be represented as a set of offerings and revenues. Here offerings are the services or goods or even both that are given to the customers by the provider. Some offerings may not be associated with revenue if they are provided to the customers for free. But for other offerings that are provided to the customers for a payment are associated with revenues. On the other hand, revenues are the total monitory income achieved by the provider. The economic unit is the unit which creates the value, and this unit may be a sub-service system by itself. In this model, offerings and revenues are represented by arcs between the participating economic entities. This model has analyzed and computed values based on partners' satisfaction and some other additional values that are created by the relationships the various partners develop in the system.

2.3.1.1 Formal model structure

The structure of this model is presented in a flow graph which includes two domains: economic entities and offerings.

The economic entities of this model are:

- An end customer which is defined as a node with zero output.
- An original producer which is defined as a node with no input—which means the producer of the raw materials for the service system
- And the set of sellers to the customers.

2.3.1.2 Offerings of particular interest

In this model, the authors have proposed offerings of particular interest like trust, risk, relationship level, relationship cost, transaction cost and production cost. The authors have thought about trust between two partners which can be built depending on the past experience. For new partners, third-party reputation and recommendation systems will influence the trust between each other. The property of trust is defined as an insurance offering that one partner is taking against future possible unreliable behavior of another partner.

Chapter 2 – Literature review

Relationship level is defined as the overall relationship quality. Trust between buyers and sellers can be enhanced by the increasing relationship level. When the relationship level between two partners is low then the risk level is high and as the relationship level improves, the risk level reduces.

Relationship cost associates with the cost for maintaining certain relationship level with the adjacent partners. For example, free service, gifts, visits, promotion campaigns can be offered to the customers from its supplier to maintain a high level of relationship. To generate more revenue, more effort is required at a higher cost. Therefore, if the current satisfaction index is low then there will be more relationship cost in the future to improve the situation.

Transaction cost such as cost for search and information, bargaining costs, and contract monitoring costs can be modeled through an offering and associated revenue. Costs acquired by these transactions are defined as transaction costs. A higher satisfaction index with a partner implies lower transaction costs as less transaction is needed.

Production costs are defined as offerings between economic entities or between entities and individuals. Labor and investments in buildings and equipment are examples of this kind of offerings used in this model.

Since the flow rates of offerings and revenues change with time, value in general can be expected to change with time. The authors have tried to capture the variability of the various flows in small and long time periods.

In order to make the business operations successful, each partner cooperates with the others. Participating in the service system is counted as valuable whereas opposed to not participating, or participating in another service system. In this paper, a quantitative estimate of this value and how this value changes with time is monitored.

The authors have considered intangible assets such as process knowledge, planning knowledge, technical knowledge, brand names, etc. which play a vital role in the creation of value in service systems. Intangible assets are those identifiable non monitory assets that cannot be physically measured, touched or seen. Intangible exchanges between partners influence properties such as satisfaction index of entities though do not directly generate revenues. For the evolution of the system these also help to create relationship within a system.

2.3.1.3 Computing value

In this paper, the value of each participating partner in the system is computed by his revenue and cost. These revenue and cost have been derived based on their income and expenditure respectively. The total value also depends on some other values like relationship level, satisfaction index which are related to the intangible asset concepts as they make some affects on revenue of each particular partner. So this value can be measured by the amount of revenue one partner gets by selling goods and services to other associated partner. The authors have given more importance to satisfaction index because from this, the values like relationship level can be easily observable. They have computed the satisfaction index of each partner within a time period. A high satisfaction index increases the revenue and relationship level and declines the risk level. In the computation, the value of each participating partner is computed based on the revenue, cost and the relationship value. Here relationship value depends on the satisfaction index of the customer and the price of the offerings they buy.

2.3.2 A Repair Service System

In this paper, the value chain of a car manufacturing repair service system has been analyzed as a case study. Here manufacturers, dealers and suppliers are participating in the system to satisfy customer requests. The value that the dealer acquires by participating in the service system depends on the annual revenue, total annual purchase and the annual average revenue depending upon the ratio of present versus past satisfaction index. The value for OEM (owners of original equipment manufacturer) depends on the total annual revenue, total annual purchase and average revenue depending upon the ratio of present versus past satisfaction index. The revenue of OEM is defined by the amount that it gets by offering parts at a fixed rate. The purchasing cost includes cost for buying parts from supply chain suppliers, warranty repairs and defective parts replacement, parts catalog content preparation and mailing and additionally help desk labors salary. The value of TPS (Third party suppliers) and SCS (Supply chain suppliers) depends on the revenue that they get by selling parts and the average revenue depending upon the ratio of present versus past satisfaction index.

The total value of repair service system depends on the values of revenues of each partner adjusted by the ratio of present versus past satisfaction indices, value of the customer and the cost of warranties paid by OEM.

2.3.3 Transformation of the Repair Service System

The conventional repair service system takes longer repair times which causes the lower satisfaction index of the car owner, thus reduces the value of the entire service system. To increase the value of the repair service system, the authors have proposed a transformation of the traditional service system to a new one in which a solution provider achieves interoperability between the information systems of the partners through a central portal created by the manufacturer. Everyone can access and get the up-to-date information about parts through this portal. As a second transformation they have proposed to replace the solution provider with an outsourcer who provides the electronic catalog system and maintenance as a service. In these structures, the authors have computed the value and they found that the repair time is reduced and customer satisfaction is increased, leading to increased sales. By this analysis, they have observed that the total value of the business is increased after the transformations as the cost of creating the information system or paying an outsourcer to provide it are lower than the catalog generation and delivery costs.

2.3.4 Relevance to our work

Through this paper we have got the idea about the e-business model of a service system and how to compute the value of each partner in a flow model which has induced us while we have done our thesis work. We have implemented this idea on a web service of an Online Book Shop. We have computed satisfaction index which adds value to the profit of an organization which concept is inspired by the intangible asset discussion of this paper. In our assumption we have included different costs such as transaction cost, offering cost, advertising cost, employee salary etc. in our computation which is presented in this paper. The authors of this paper have transformed the conventional repair service system by introducing a central portal and replacing the solution provider whereas we are using Retry and Substitution as repair strategies to reduce unsuccessful requests. The goal of this paper is to analyze the service system in terms of economic value whereas our goal is to analyze how the repair actions can increase the economic value of a web service which includes both profit and customer satisfaction. This paper shows how to improve customer satisfaction of a Car Manufacturer Company and provide options and analysis models for outsourcing decision makers whereas our work applies repair actions to an Online Book Shop to reduce unsuccessful requests and analyze how to improve the worst situation by applying these repair actions.

CHAPTER 3

Analysis of an e-business scenario

The technology of web services has given a huge contribution in e-business field. Business organizations are now adapting web services to provide better services to their customers. Adapting web services gives benefit to the organization also as building and maintaining web applications are now easier and much more cost effective. For this reason, organizations are getting more profitable economy. In our thesis, we have proposed an e-business scenario. In this scenario, different entities have adopted web services in their applications. As our main attention is the self repairability of web services, we have tried to find out what are the general reasons that cause failures of web services. In this chapter, we have mentioned some faults that may occur in our proposed scenario. Then we have discussed some repair actions which can be applied during the failure of a web service. Details about how these repair actions will be executed can be found in [9] where a framework has been provided which support the self healing execution of complex web services. In our thesis we are mainly concentrating on estimating value from business perspective after applying repair actions in the web services and also we are trying to compare between different repair actions from business perspective. For this reason, we have tried to find out the faults and failures which can occur in this business scenario upon which the repair actions will be executed.

In Section 3.1, we describe our proposed e-business scenario. Here we introduce the business entities who are associated with this scenario. We also discuss the relationship between the entities. In Section 3.2, the business execution process of the entities is presented through some scenarios. In Section 3.3, we provide some general faults and failures that may occur in this scenario. Here we consider these faults and failures from the perspective of our proposed e-business model. In Section 3.4, a hypothesis of repair strategies is discussed which can be applied in case of failure of a web service.

3.1 E-Business scenario

In our proposed e-business scenario, the central business entity is the owner of the Online Book Shop. For simplicity, we have mentioned this entity as OBS. It is a simple web based application in which people can find different books from different publishers and buy books according to their choice. When the customer gives an order of book to OBS web site, OBS sends this order to the corresponding publisher. If the book is available in the publisher's inventory, the book(s) is delivered to the customer. In Fig 3.1, the model has been presented. This model shows all the entities who are related to this whole e-business and the relationship between these entities. In the Fig, the nodes represent the business entities and the arcs represent the action performed by one entity with another entity. The entities are:

1. OBS (Owner of Online Book Shop)
2. Customer
3. Publisher
4. Inventory
5. Help desk
6. Book packager
7. Advertisement manager
8. Advertisement containers

Chapter 3 – Analysis of an e-business scenario

OBS: Owner of Online Book Shop (OBS) is the main central business entity of the scenario. This entity is responsible for giving the free web portal to the customers. From this web portal, customers can find their desired books from different publishers and can buy accordingly. OBS maintains the records of all registered publishers and all books of these publishers in its database.

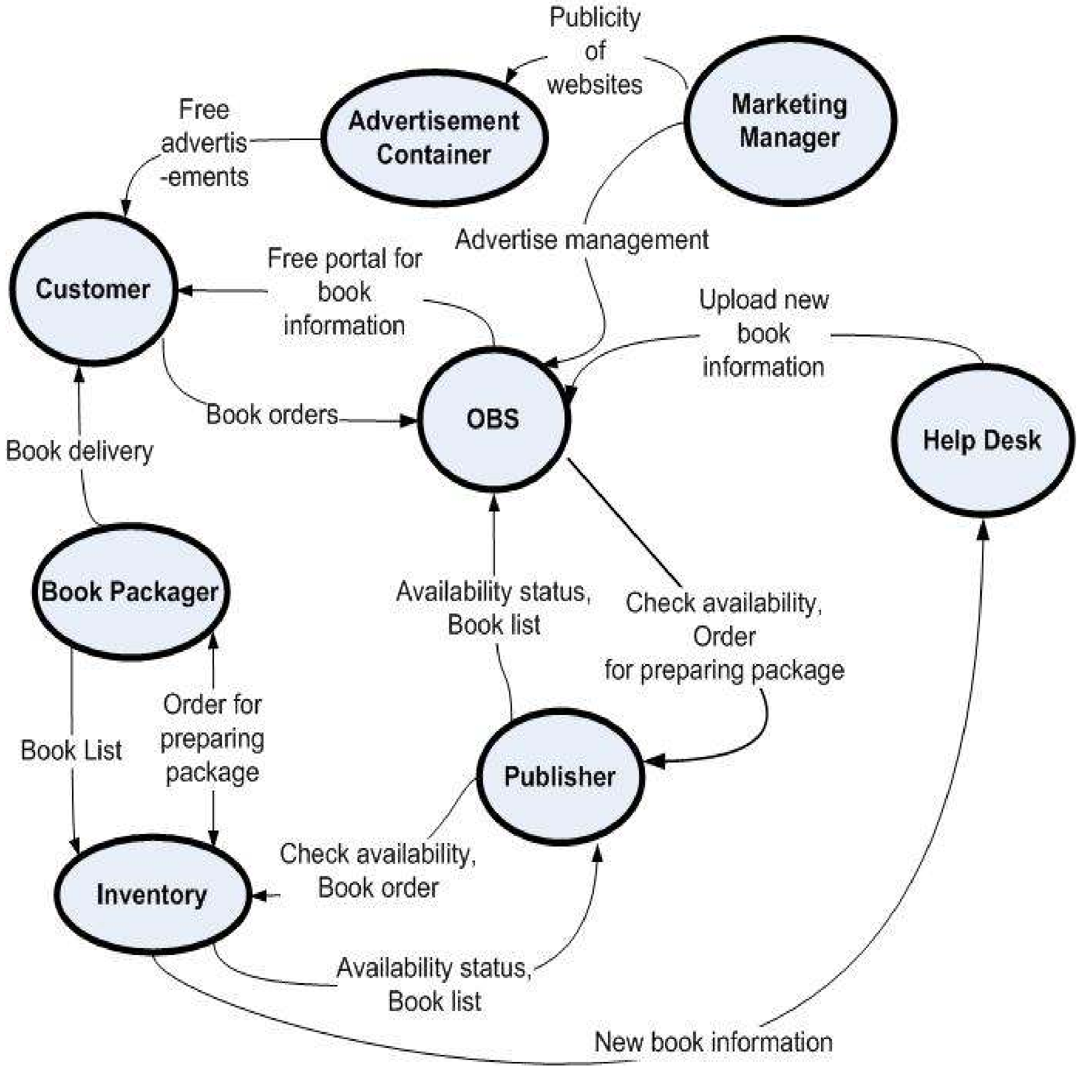


Fig 3.1: An e-business scenario

Customer: The node Customer represents all the customers who buy books from OBS. They can see all the published books from different publishers from this web portal. From the list of published books, they can choose their desired publisher's books.

Publisher: Publisher represents the set of publishers who are registered with the OBS. The publishers can sell their books through this web site of OBS. Each publisher provides the names of all books along with the writer's name to the OBS. The list of all books of each publication house is shown to the customers through the website of OBS.

Inventory: Each publisher has his own inventory of books. A database is managed to store the information of all books in each inventory.

Help Desk: Help Desk is acting a role as a mediator between the OBS and inventory of each publisher. The purpose of Help Desk is to get information of all the books that have been published by each publisher form the inventories and provide this information to the OBS. Help Desk performs its operation on daily basis.

Book Packager: Book Packager works under each inventory. Upon receiving the orders of books from the inventory, Book Packager makes packages of books and delivers those packages to the respective customers. Book packaging is the activity which can be carried out by human.

Marketing Manager: Marketing Manager handles the marketing department of OBS. This entity is responsible for advertising OBS in other websites. It also handles the advertisements of other companies in the website of OBS.

Advertisement Container: Advertisement Container is the set of all websites into which the advertisements of OBS are shown. From these containers, both customers and publishers can get the information about OBS.

3.1.1 Transactions between entities

Each business entity has adopted web service in this scenario. The communication between entities means the communication between the web services of the entities. OBS contains only the names of the published books of a publishing house. It doesn't have the availability information of the books. So, when the orders come to the OBS, OBS asks the respective publisher about the availability of the ordered book(s) (Fig: 3.1). Publisher queries to the inventory and gets the availability status of the ordered books. Here query defines asking the book availability to the entity. Publisher then sends this status to the OBS to inform that whether the book is available or not. OBS uploads the status on the website so that Customer can see the availability status of the book(s). If the book is available then OBS sends the purchase order to the publisher. This order goes to the Book Packager entity through the Publisher and Inventory entities. Book Packager makes packages of the books. For final confirmation before sending the books to the customer, Book Packager makes a list of packaged books and sends this list to the Inventory. Inventory sends this book list to the OBS through the Publisher. From the web portal of OBS, customer can verify whether the list of books is correct or not. After getting the confirmation from the customer, Book Packager sends the book(s) to the customer. But if the book is not available, then the customer will be informed about this unavailability through the OBS.

3.2 Business process execution

In this section, we have shown the execution of business process between Customer, OBS, Publisher, Inventory and Book Packager. The executions have been shown by the following three scenarios where all successful and unsuccessful paths have been taken into account. In the first scenario, we have assumed that the executions of business processes are successful i.e. no failure has been occurred in web services. In second and third scenarios, Customer does not get his/her desired book(s) due to some failures of web services.

In Fig 3.2, first scenario has been shown. This scenario shows successful path between the entities. In the database of OBS, the names of all books of all registered publishers are stored. Each book has its own unique id. After getting the order from Customer which contains only the book name of a specific publisher, OBS sends the corresponding book id to the respective Publisher. Publisher then finds the book id from his database which matches the ordered book id. Then he sends the name and id of the ordered book to the Inventory to check for availability. Inventory sends the availability status to the publisher and this status is shown to the Customer through the website of OBS. When Customer confirms the order, the order for packaging book is sent to the Book Packager through the entities. Book Packager makes the list of ordered books and this list has been sent to the Customer for the final confirmation. If the confirmation is OK, the package of ordered book(s) is delivered to the Customer by the Book Packager. In Fig 3.3, we have shown the second scenario when book is not available i.e. the availability status is NO. After getting this availability status, customer cancels the order. In Fig 3.4, the third scenario has been presented where the list of books is incorrect which has been provided from the corresponding Publisher to the Customer through the OBS website. Customer confirms the list is

NOT OK and cancels the order. These failures and the reasons behind these failures have been explained in the next section.

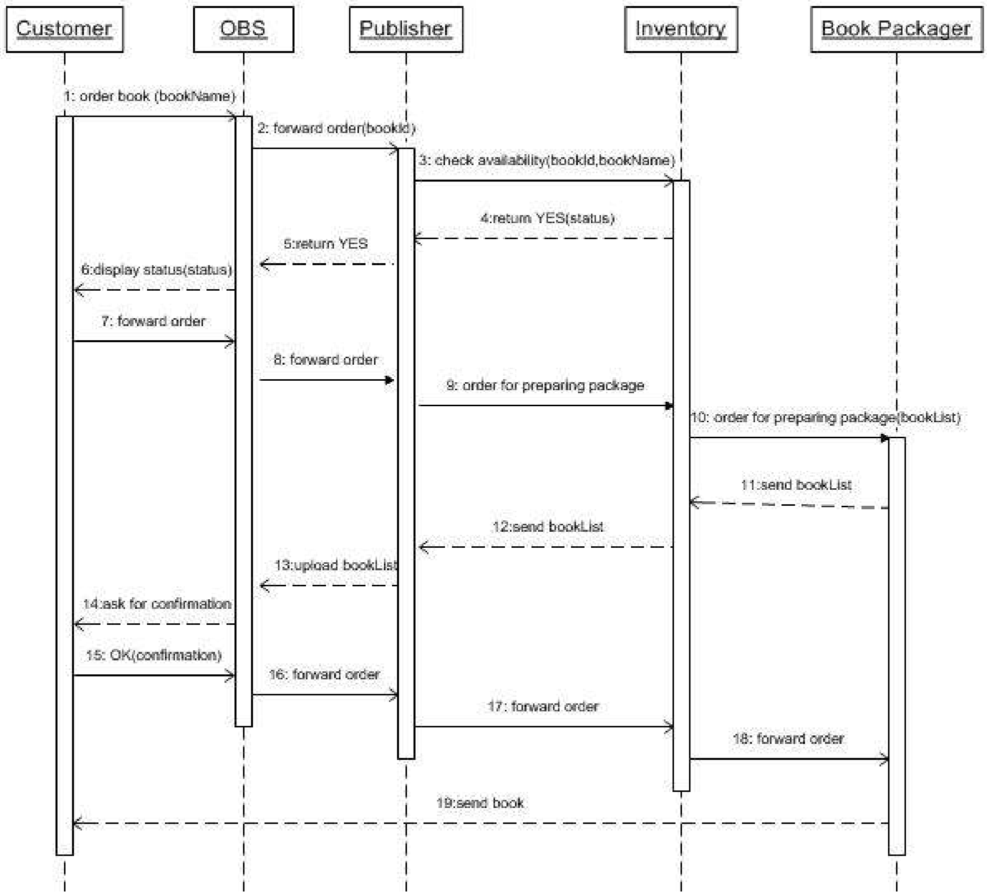


Fig 3.2: Scenario 1 of business process execution

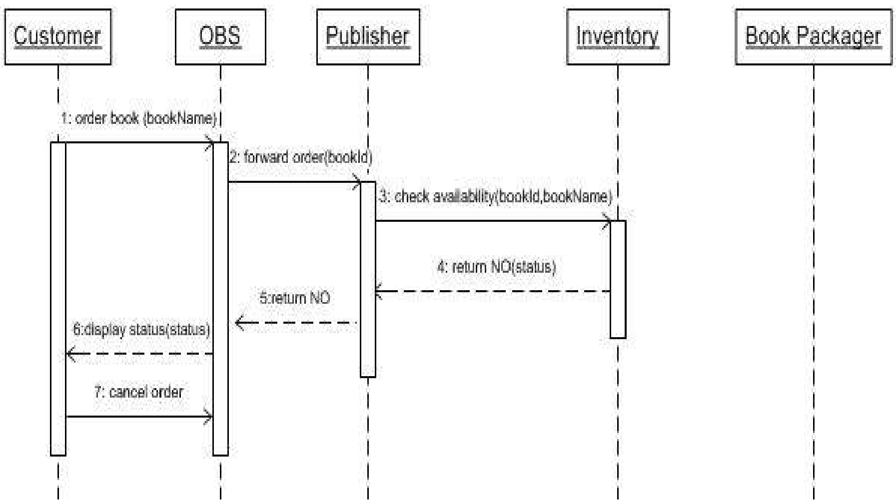


Fig 3.3: Scenario 2 of business process execution

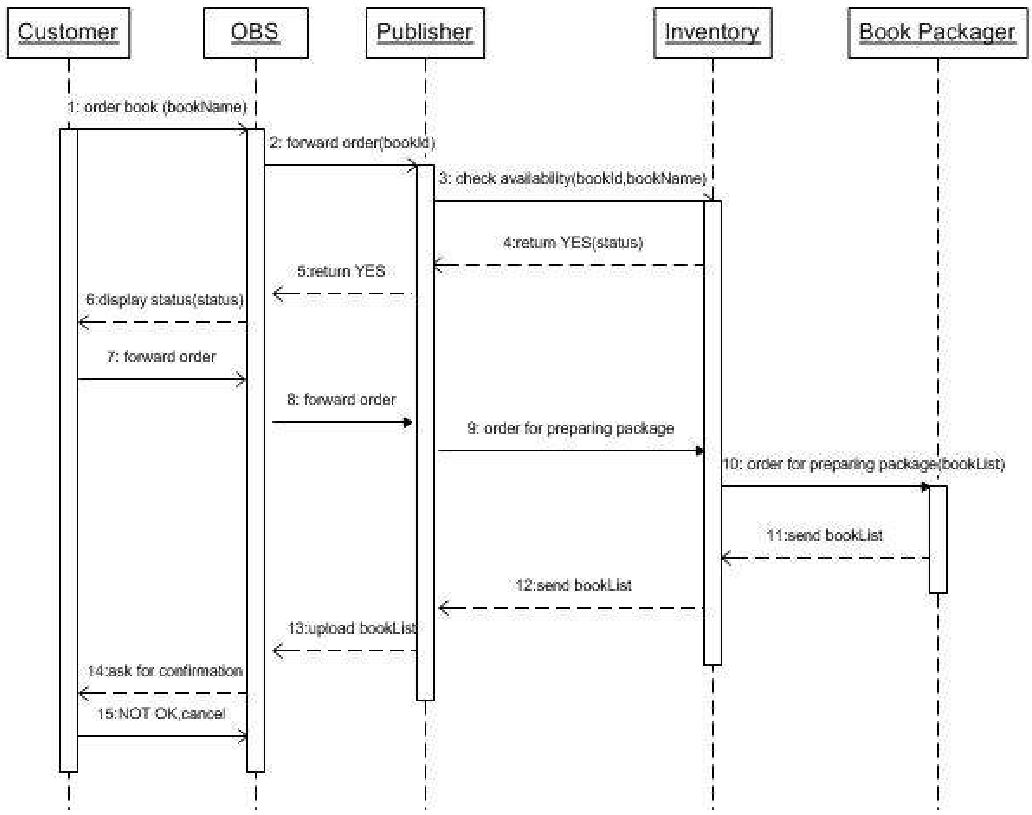


Fig 3.4: Scenario 3 of business process execution

3.3 Faults and Failures

As we mentioned earlier, faults are the reasons that cause failures of web services. Here failure of web service mainly indicates wrong information provided by the service. In Fig 3.3, we saw that sometimes OBS cannot give the ordered books to the customer as the books are not available. This is one situation in which the books are actually unavailable. But it may happen that OBS is getting wrong information from the publisher about the availability of the books i.e. books are actually available but due to some internal errors or faults, a wrong information from the publisher is generated. Also in Fig 3.4, we saw that sometimes Book Packager produces wrong list of books. For this reason, we have tried to find out the general faults and failures of web services so that we can apply some repair actions to repair the faulty services. In this section, we shown some failures of web services and try to find out the faults that generate these failures.

According to [9], each fault is represented as faulty service and faulty result i.e. a fault represents which the faulty service is and what result the faulty service is producing. In the following we have shown faults that may occur in the execution of the above business process.

Fault1: <OBS, bookId> {the corresponding bookId is wrong in the OBS database}

Fault2: <Publisher, bookName> {there is an inconsistency between bookId and bookName in the Publisher's database}

Fault3: <Inventory, availability status> {in the database of Inventory, the availability status information is wrong}

Fault4: <Book Packager, booklist> {wrong booklist has been produced by the Book Packager}

If we analyze Figs 3.3 and 3.4, we can see that there can be two types of failures in the web services. In Fig 3.3, the Publisher confirms that the book is not available. This availability information of requested book(s) can be wrong due to Fault1, Fault2 or Fault3. On the other hand, in Fig 3.4, the book list is incorrect which is provided by the Publisher through OBS. This wrong book list can also be generated from Fault1, Fault2, Fault3 or Fault4. In Table 3.1, the failures and the corresponding faults have been shown.

Failures	Faults
Fail1: Book order results wrong information from Publisher (book available/ unavailable)	Fault1 Fault2 Fault3
Fail2: Book order results wrong information from Book Packager (wrong book list)	Fault1 Fault2 Fault3 Fault4

Table 3.1: Failures and faults

3.4 Repair actions

In our thesis, we have considered two types of repair actions which will be applied during the failure of a web service. They are retry and substitution. In this section, we discuss elaborately about the two repair actions and apply these repair actions to the faulty services.

Retry: Retry is the invocation of the unavailable services until they are available [10]. When OBS gets the book unavailability status from the Publisher (Fail1), the retry action will be applied with the same book order. We have seen in the previous section that this wrong information (Fail1) can be generated due to Fault1, Fault2 or Fault3 (Table 3.1). So when OBS sends the book order again. Each of the participating entity including OBS itself checks their databases and fixes the problem if any error is found. In case of Fail2 (Table 3.1), the retry action is again applied with the same book order so that database error can be fixed (if any) and Book Packager can recheck the book list. Normally, if the error is in the book list, after applying retry action the Book Packager can fix this problem. But if the problem exists in the database of any entity, then retry may have to be applied several times to get the right information. Retry can be applied several times but the actual number of times retry will be applied depends on the execution cost of each retry, customer satisfaction and other factors of the organization.

Substitution: When a service is considered as completely unavailable, then the failed service is substituted with another service. After applying retry for one or several times, if OBS gets still the wrong information, then it will be considered that book is actually unavailable or the internal error cannot be repaired. Then OBS will substitute that publisher with other one. For substitution, OBS queries the ordered bookName to one or more publishers at a time to know if the book is available with the corresponding publisher. The publishers who have similar types of books, response with the book names and OBS asks Customer to choose book from those options. Again how many queries OBS will do depends on the execution cost of each query, customer satisfaction etc.

In Fig 3.5, we have shown Fail1 and the associated repair actions. At the first query to the Publisher, if the status is unavailable, OBS applies retry action for n times. If the status is available, then the process executes in the normal flow. But if the status is still unavailable then OBS applies substitution. For substitution, OBS sends book search queries to m number of different publishers.

In Fig 3.6, Fail2 and the corresponding applied repair actions have been shown. When the customer says the book list is not ok which is sent by the Book Packager, a message comes to the Customer requesting for what for a while. As the book list can be faulty because of Fault1, Fault2, Fault3 or Fault4, OBS applies retry action for n times so that each entity can verify whether there is any error in its database. After that if the booklist is still erroneous, then OBS applies substitution with m number of Publishers. After applying repair actions to the both failures, if the web service is still faulty, then the order from the Customer will be discarded.

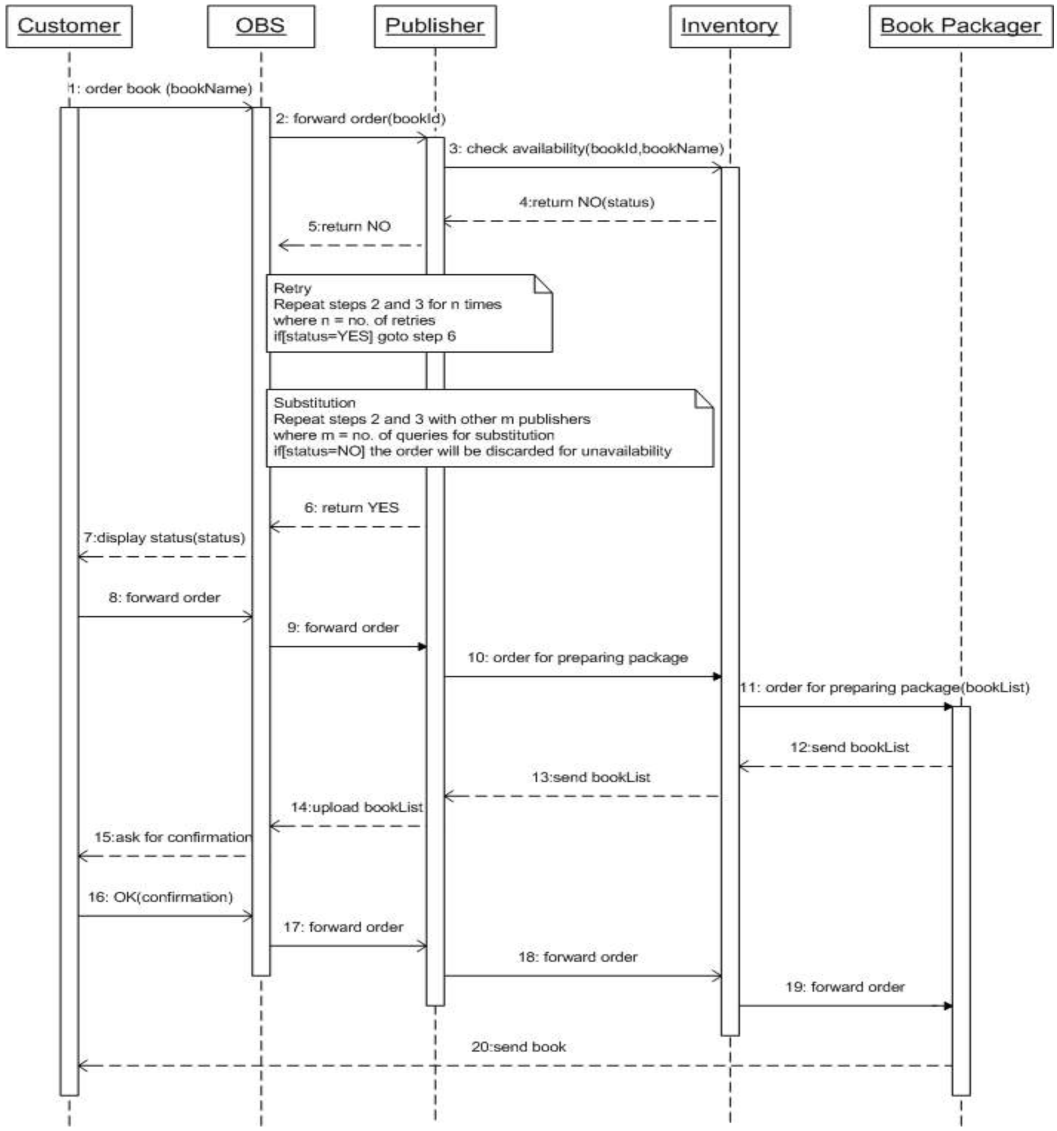


Fig 3.5: Repair actions applied to Fail1

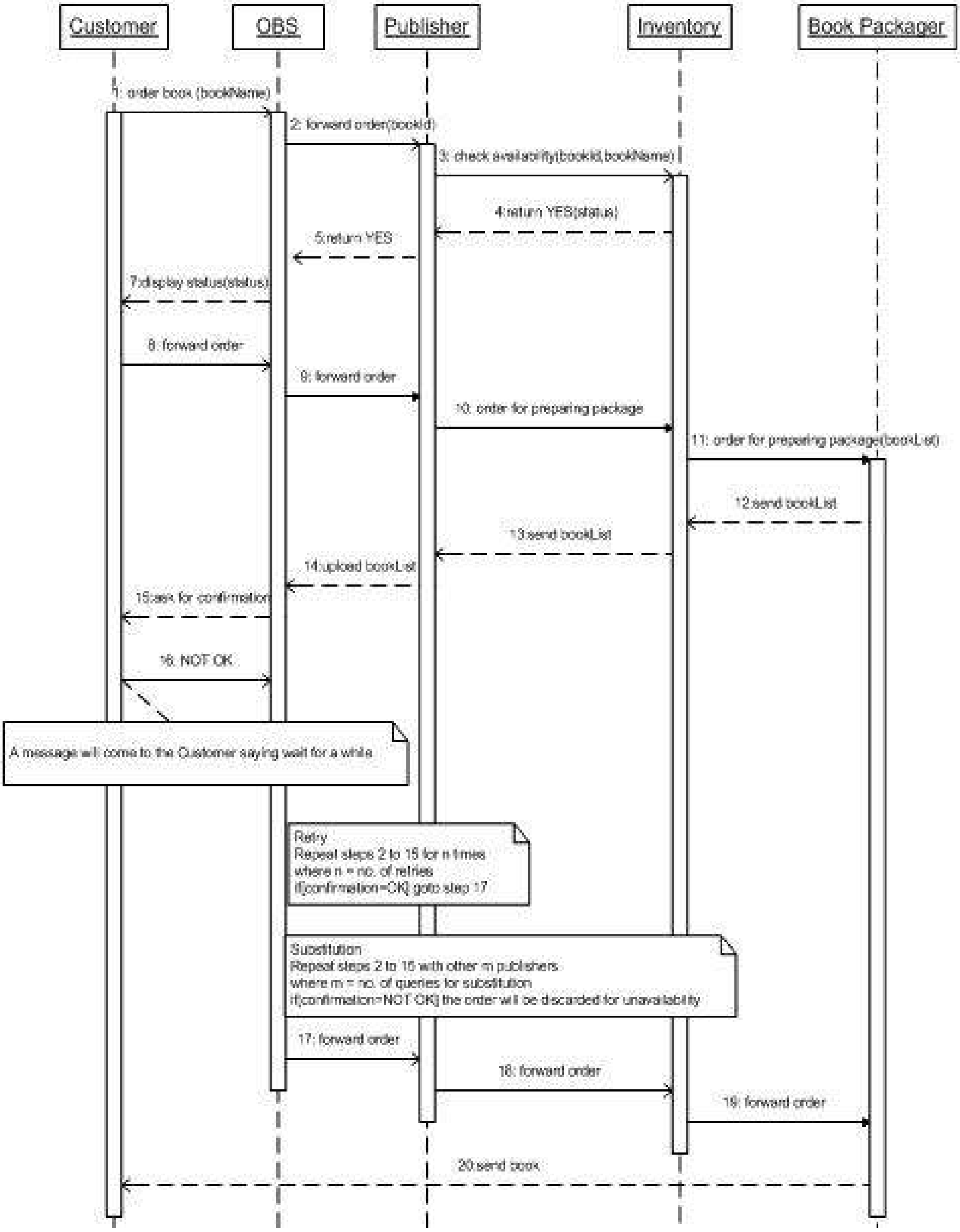


Fig 3.6: Repair actions applied to Fail2

Chapter 3 – Analysis of an e-business scenario

In this chapter, we have proposed an e-business scenario. Our work goes further considering this scenario. In this scenario, we have talked about some faults that produce the failures in web services. We have also proposed a hypothesis of applying the combination of retry and substitution during the web service failures. According to this hypothesis, substitution is applied after applying a number of retries. But in our analysis our main goal is to compute the economic value of the organization by applying retry and substitution separately, independent of each other so that we can understand which one will be beneficent for the organization. This analysis can be extended further by computing the economic value considering the combination of retry and substitution.

CHAPTER 4

Computation of economic values

In business execution process, each business entity focuses on how to maximize the profit and satisfaction of customers. Here customer is the adjacent node of the entity which demands some requirements from that entity. The revenue of a business organization combines the profit and satisfaction of customers. Different business entities are associating with our proposed e-business scenario. Each entity is considered as web service enabled. The repair actions are applied to the web services in case when services cannot fulfill the requirements of the adjacent node. As we have mentioned earlier, the whole business process execution is represented by a scenario in which the nodes are the business entities and arcs are the actions performed between a pair of entities (Fig 3.1). The purpose of our thesis is to compute the economic values of business entities affected by the repairability action of web services. For this, we have taken OBS, the central business entity for the value computation.

We have computed the economic values of OBS and have tried to find out what are the affects in economic values after applying repair actions in the failed web services applied from

OBS. To do this, we have considered a worst case of the scenario when the failures of web services occur. Then after applying retry and substitution, we have compared the changes in values between all the cases.

In computing the economic values of OBS, we have considered some business factors and properties of business entities which contribute to the economic values of OBS. In this chapter, we elaborately describe those factors and properties. We also analyze the change in values after applying the repair actions through this chapter. In Section 4.1, we discuss the business factors, properties of entities and their computations in finding the economic value. In Section 4.2, the computation of economic values is provided in the worst case of the scenario. Sections 4.3 and 4.4 discuss the computation of values after applying retry and substitution respectively.

4.1 Computations of business factors, properties and value

In this e-business scenario, we have considered mainly two business factors which contribute on the economic values of OBS. They are cost and revenue. The property of entities which we are considering is satisfaction index. In computation, we only focus on the satisfaction index of Customer and Publisher as their satisfaction is the main concerned issue for OBS. As our main goal is to analyze the web service repairability, we consider a scenario when web service failures occur. We call this scenario as worst case scenario where OBS is failed to provide service to the Customer i.e. OBS is unable to provide the required books to the Customer. According to Chapter 3, this unavailability of books is considered as Fail1 or Fail2 for which we have applied the repair actions. We have applied retry and substitution separately on the worst case to repair the services. As a result, we can compare the affects in economic values of OBS from the perspective of each repair action.

So, at first we have computed the economic value of OBS in worst case scenario. Then we have computed the value after applying retry and substitution separately. In the following sub sections, we provide the computations of cost, revenue, satisfaction and economic value of OBS.

Chapter 4 – Computation of economic values

4.1.1 Computation of cost

Cost is considered as the expenditure for different business operations. Here business operations represent providing and purchasing services, activities and maintenance. In our analysis, the cost of the OBS is the amount of money which it has to spend for giving services to its customers. The major fields at which cost is associated are database management, marketing management, help desk, offerings sent to the publishers and transactions for queries and website domain. Database management cost is associated with the salary of the database administrator. Marketing management cost is associated with advertisement cost and marketing manager's salary. There is a fixed rate of money for advertising the website through other websites which OBS has to pay. OBS also pays salary to the help desk for its service. Average offering charge is referred to the cost for sending information through mails, newsletters, catalogs etc. to the publishers. Transaction cost supervising the cost of exchanging information between partners. OBS has to pay for the domain of the website also in yearly basis. Each year it has to pay a fixed amount for the domain cost.

Considering the total number of customers, publishers, requests, advertising containers, and help desk salary per month the cost becomes:

Cost of OBS per month, C_w = database management cost + no. of publishers * average offering charge + no. of requests per month * transaction cost per request + no. of advertisement containers (websites) per month * advertisement cost for each website + no. of help desk employees per month * salary of each help desk employee + marketing manager's salary.

For simplicity, we have shown these terms in notations.

$C_w = D_c + N_p * f + N_t * C_t + M * a_d + l_h * r_h + r_m$

Cost of OBS per year, $C_{wy} = W_c$ (yearly domain cost) + 12 * C_w

Here, D_c = database manager's salary,

N_p = no. of publishers,

f = average offering charge,

N_t = no. of requests per month,

C_t = transaction cost per request,

M = no. of advertisement containers (websites) per month,

a_d = average advertisement cost spent for each container (website) ,

l_h = no. of help desk employees per month,

r_h = salary of each help desk employee,

r_m = marketing manager's salary per month ,

W_c = website domain cost.

As we have taken into consideration that each request has a transaction cost, total transaction cost can be achieved by considering total number of requests per month. Cost for the advertisements of the website of OBS can vary in different containers depending upon size of the content, so we have taken the average cost for each container. We have assumed that, there will be only one advertisement in each website.

4.1.2 Computation of revenue

For a business organization, revenue is the total amount of money received by the organization for goods sold or the services provided during a certain period of time. In our thesis, revenue is considered as the earnings of the OBS from different entities. The business of OBS runs through the investment of this revenue. OBS earns this revenue by the monthly registration fees from the publishers and additionally the amount which it gets from advertising other websites in its own website.

Considering the total number of publishers and advertising websites, the earning of OBS is as follows:

Revenue of OBS per month, R_w = no. of publishers * monthly registration fee + no. of advertising websites * average earning from each website.

For ease of computation, we have shown these terms in different notations.

$R_w = N_p * a + N_a * P_w$

Annual revenue of OBS, $R_{wy} = 12 * R_w$

Here, N_p = no. of publishers,

a = monthly registration fee,

N_a = no. of advertising websites,

P_w = average earning from each website

In our computation, again we have taken the average of earning from each advertising website because advertisement cost may differ depending upon the size and content of the advertisement. The monthly registration fee is same for all publishers.

4.1.3 Computation of satisfaction index

Satisfaction index is the property of the entities which indicates the satisfaction level of those entities. As Customer and Publisher both are connected to the OBS, their satisfaction about OBS is very important because it adds value to the revenue of OBS. In our thesis, this satisfaction is represented by an index which is associated with availability, relationship level and risk level between the connected entities. Availability indicates whether the book is available or not. If the customer gets his desired books from OBS, the availability is 1 otherwise 0. On the other hand, if publisher can provide the book to the customer through OBS, then availability is 1 otherwise 0. Relationship level and risk level between a pair of entities are highly dependent on availability. From customer's point of view, if the availability is high then relationship level with OBS will also be high lowering the risk level and vice versa. So the total satisfaction index of customer about OBS will be affected by the changes of these properties. From the publisher's point of view, if he cannot provide the book to the OBS, then risk level will be high between publisher and OBS lowering the relationship level between them and vice versa. As a result, the satisfaction index of publisher about OBS will be low or high depending on the changes of availability, relationship level and risk level. Here, Relationship level indicates the level of relationship between the connected entities. Relationship level is 1 if the relation level between two entities is high, 0.5 if the relationship is medium and 0 if the relationship is low. Risk level provides the level of reliability between the entities. If the risk level is 1 it indicates that the risk in operating business between two partners is high and 0.5 indicates low. Here we have considered that there is always some risk in doing business, thus the risk level cannot be 0. The general equation for satisfaction index is:

$$S = I_{av} * \text{Availability} + I_{rel} * \text{Relationship level} + I_{risk} \div \text{Risk level}$$

Where, $I_{av,}$ I_{rel} and I_{risk} are the importance of availability, relationship level and risk level respectively. Importance level is 1 when the importance of the particular property is high, its 0.5 when importance is medium and its 0 when the importance is low. We have considered the level of importance from both the customer and the publisher's point of view. Table 4.1 shows the level of properties and Table 4.2 shows the level of importance whereas Table 4.3 shows the importance level from customer's and publisher's perspective.

Level of importance	Low	Medium	High
Availability	0	-	1
Relationship level	0	0.5	1
Risk level	0.5	-	1

Table 4.1: Level of properties

Level of importance	Low	Medium	High
Availability	0	0.5	1
Relationship level	0	0.5	1
Risk level	0	0.5	1

Table 4.2: Level of importance

Level of importance	Customer	Publisher
Availability	1	0.5
Relationship level	0.5	1
Risk level	0.5	0.5

Table 4.3: Level of importance from the customer's and publisher's perspective

From the customer's perspective, the importance of availability is highest among other properties because availability of books is the most important property for the customer's satisfaction. Customers give more importance to the fact that whether their desired books are available or not. Whereas the importance of availability of a publisher is medium because their intention is to supply more books and it is not necessary to have all the requested books available. The importance of relationship level is medium considering the satisfaction of customer because customers have other options to buy books from other shops. On the other hand, it is high for a publisher because a publisher should have good relationship with OBS to increase sales. The publisher, who has good relationship with OBS, gets some preferences because OBS choose that publisher in case of book unavailability. There are always some risks before any business operation takes place between the entities. That's why, the importance of risk of a customer and a publisher is medium because it does not affect much on the satisfaction index of both of them.

We have computed the satisfaction index of a customer and a publisher about OBS separately considering different sub properties, then computed the satisfaction index of all the customers and publishers and finally added them for getting total satisfaction index of all customers and publishers about OBS.

Our analysis of satisfaction includes the initial satisfaction index when the business has just started. Then we have considered a worst case where the number of unsuccessful requests is high and computed the satisfaction index in this scenario. We have applied retry and substitution as a repair strategy to the worst case scenario and computed the satisfaction index after applying these repair actions.

4.1.3.1 Computation of satisfaction index at the initial level

Initially the availability is high for both customer and publisher because we think that books are available at the initial level. As there is no transaction of books occurred at the beginning, we have assumed that the relationship level of customer with OBS is low and with publisher is medium as the publishers have already registered in this website. So there is some relation between OBS and publisher. The risk level is high for both of them as the customer is placing order in this website without knowing much information about book availability and the publisher is doing business for the first time with OBS. We have computed the satisfaction index of the customers and the publishers by using the following data:

Properties	Level of importance	Level of property
Availability	1	1
Relationship level	0.5	0
Risk level	0.5	1

Table 4.4: Level of importance and properties at the initial level from customer's perspective

So, satisfaction index of a customer at the initial level is:

$S_c = 1 * 1 + 0.5 * 0 + 0.5 \div 1 = 1.5$

Properties	Level of importance	Level of property
Availability	0.5	1
Relationship level	1	0.5
Risk level	0.5	1

Table 4.5: Level of importance and properties at the initial level from publisher's perspective

Satisfaction index of a publisher at the initial level (using Table 4.5) is:

$S_p = 0.5 * 1 + 1 * 0.5 + 0.5 \div 1 = 1.5$

We have assumed that, at the initial level, all the customers have same satisfaction index (S_c) and all the publishers have same satisfaction index (S_p). Thus, the total satisfaction index of OBS at the initial level is:

$N_c * S_c + N_p * S_p$

Here, N_c = Number of customers,

S_c = satisfaction index of the customers,

N_p = number of publishers,

S_p = satisfaction index of the publishers.

4.1.4 Computation of economic value

Our analysis is aimed to compute economic value of OBS and to observe the changes in value according to the changes in different parameters. The cost and revenue that we have already computed usually gives the idea about the profit of an organization. In computation, the value of cost and revenue are fixed in different situations (worst case or after applying repair actions). But the satisfaction index changes depending on the situation because after applying repair actions, it affects the satisfaction of customer and publisher. In our analysis, we have considered satisfaction index as an important property which adds some value to the revenue. We have tried to find out the changes in satisfaction index in different cases. For that, we have divided the satisfaction index in different cases by the initial satisfaction index so that we can get the ratio between them. This ratio actually indicates the change in satisfaction index from the initial one in different cases. Then this ratio is converted to a monitory value by multiplying with the revenue of OBS. Afterwards, it is added with the difference of cost and revenue. The value of

OBS is shown below:

Value of the OBS, $V_w = R_{wy} - C_{wy} + (Sat_c \div Sat_i) * R_{wy}$

Here, R_{wy} = Annual revenue of OBS,

C_{wy} = Annual cost of OBS,

Sat_c = Satisfaction index in different cases (worst case, after retry and after substitution),

Sat_i = Satisfaction index at the initial level.

In the following sections, we compute the economic value of OBS in worst case scenario. Then the value is computed again after applying retry and substitution respectively.

4.2 Value computation in worst case scenario

In Section 4.1 we have mentioned that we consider a worst case scenario in which the number of failures is high i.e. a number of customers are not getting their desired books for some failures of web services. Thus the satisfaction of customers decreases and the satisfaction of publishers who could not sell their books also decreases. As a result, the value of OBS decreases. To improve this situation, the repair actions have been applied. In this section, we compute the value of OBS in the worst case scenario.

In the worst case scenario, the cost and revenue remain same as the failure of web services i.e. book unavailability does not affect these factors. They are the fixed values for OBS in this scenario. The only thing that will change is the satisfaction index of OBS from the perspective of customer and publisher because book unavailability affects both the customer and the publisher. So, at first we have computed the satisfaction index of OBS in the worst case scenario and then we have computed the economic value of OBS.

4.2.1 Satisfaction index in worst case

In the worst case scenario, we have divided the customers into two groups: The customers who did not get their desired books and the remaining customers. The publishers are also divided into two groups: one with the publishers who could not fulfill the orders and other with the remaining publishers. In Section 4.1.3, we have mentioned the general equation for satisfaction index which is:

$S = I_{av} * \text{Availability} + I_{rel} * \text{Relationship level} + I_{risk} \div \text{Risk level}$

For the customers who do not get the book, the availability is 0, relationship level is 0 and risk level is 1. That means this is the worst possible situation which we are considering. This is shown in Table 4.6.

Properties	Level of importance	Level of property
Availability	1	0
Relationship level	0.5	0
Risk level	0.5	1

Table 4.6: Level of importance and properties in the worst case from unsatisfied customer's perspective

So, the satisfaction index of the unsatisfied customers is:

$S_{C\text{-uns}} = 1 * 0 + 0.5 * 0 + 0.5 \div 1 = 0.5$

The values of properties to compute satisfaction index of other customers remain same as initial. Table 4.7 provides level of importance and properties in the worst case from other customer's perspective.

Properties	Level of importance	Level of property
Availability	1	1
Relationship level	0.5	0
Risk level	0.5	1

Table 4.7: Level of importance and properties in the worst case from other customer's perspective

The satisfaction index of other customers which does not change is:

$S_{C\text{-others}} = 1 * 1 + 0.5 * 0 + 0.5 \div 1 = 1.5$

In case of Satisfaction index of the publishers who fail to provide the required books by OBS, the availability is 0. As a result, the relationship level also becomes 0 and the risk level becomes 1 (Table 4.8).

The satisfaction index of the unsuccessful publishers is:

$S_{P\text{- uns}} = 0.5 * 0 + 1 * 0 + 0.5 \div 1 = 0.5$

Properties	Level of importance	Level of property
Availability	0.5	0
Relationship level	1	0
Risk level	0.5	1

Table 4.8: Level of importance and properties in the worst case from unsuccessful publisher's perspective

The values of properties to compute satisfaction index of other publishers remain same as initial (Table 4.9).

Properties	Level of importance	Level of property
Availability	0.5	1
Relationship level	1	0.5
Risk level	0.5	1

Table 4.9: Level of importance and properties in the worst case from other publisher's Perspective

Satisfaction index of other publishers is:

$S_{P\text{-others}} = 0.5 * 1 + 1 * 0.5 + 0.5 \div 1 = 1.5$

So, the total satisfaction index in the worst case is:

$N_{C\text{-uns}} * S_{C\text{-uns}} + N_{C\text{-others}} * S_{C\text{-others}} + N_{P\text{-uns}} * S_{P\text{- uns}} + N_{P\text{-others}} * S_{P\text{-others}}$

Here, $N_{C\text{-uns}}$ = no. of unsatisfied customers,

$S_{C\text{-uns}}$ = satisfaction index of unsatisfied customer,

$N_{C\text{-others}}$ = no. of remaining customers,

$S_{C\text{-others}}$ = satisfaction index of remaining customers,

$N_{P\text{-uns}}$ = no. of unsuccessful publishers,

$S_{P\text{- uns}}$ = satisfaction index of unsuccessful publishers,

$N_{P\text{-others}}$ = no. of remaining publishers,

$S_{P\text{-others}}$ = satisfaction index of remaining publishers.

4.2.2 Economic value in worst case

In the worst case scenario, as the satisfaction index of both customers and publishers decreases, the economic value of OBS also decreases. It is obvious that if the ratio of worst case satisfaction index and initial satisfaction index is negligible then it would not add much value to the revenue while if the ratio is high then it would add some significant value to the revenue. In the worst case the equation of the value would be as follow:

Value of the OBS in the worst case, $V_{worst} = R_{wy} - C_{wy} + (Sat_w \div Sat_i) * R_{wy}$

Here, R_{wy} = annual revenue of the OBS,

C_{wy} = annual cost of the OBS,

Sat_w = satisfaction index in the worst case,

Sat_i = satisfaction index at the initial level.

4.3 Value computation after applying repair actions

In this e-business scenario, we have applied two repair strategies which are retry and substitution to recover from failures. For each unsuccessful request we have applied retry or substitution depending upon the fault. As mentioned in Section 3.4, retry is applied when there is unavailability of books from a specific publisher or there is a mistake in book delivery. These problems can occur due to database error, sending wrong information or human error in packaging books. If the problem still exists after applying retry several times, it is assumed that the book is actually unavailable or the fault is permanent which cannot be repaired. In that situation, substitution is applied. For substitution, OBS sends some query to the other publishers to know whether the requested book is available or not. One of them among the publishers who give responses will get the order according to the customer's choice. We have assumed that these repair actions will satisfy more customers and publishers which will affect the revenue of the OBS. We have computed the value after applying these repair actions which gives us an idea about how the value changes with each repair action. We have also assumed that the revenue is same as before but the cost and value changes after applying these repair actions. In the following discussion, we show our computation of cost and value after applying retry and substitution.

4.3.1 Computation of value after applying Retry

After applying retry, the expenditure of the OBS increases as each retry action adds some cost to the existing cost, revenue remains same but the satisfaction index changes. This change in satisfaction index affects the revenue which changes the value.

4.3.1.1 Cost after applying retry

While computing the cost after applying retry, we have assumed that for each unsuccessful request one or more retries can be applied. Number of retries per unsuccessful request is multiplied by the average transaction cost per try to get the retry cost per unsuccessful request. To get the retry cost per month we multiply this cost with the number of unsuccessful requests and added with the monthly cost of the OBS. Then we computed the yearly retry cost by adding yearly domain cost.

Retry cost per unsuccessful request, $C_{ru} = N_{rpu} * C_t$

Retry cost per month, $C_{wr} = C_w + N_u * C_{ru}$

Yearly cost after retry, $C_{wry} = W_c + 12 * C_{wr}$

Where, N_{rpu} = no. of retry per unsuccessful request,

C_t = average transaction cost per try,

C_w = cost of OBS per month,

N_u = no. of unsuccessful request per month.

4.3.1.2 Revenue after applying retry

We have assumed that the revenue remains same as before though after applying retry, the satisfaction of the adjacent partners increases which add some value to the revenue of the OBS.

4.3.1.3 Satisfaction index after applying retry

While computing the satisfaction index after applying retry, we have divided the customers into two groups as before: the customers who are getting books and other customers who are having the same satisfaction index as the initial level. We have also divided the publishers into two groups: the previously unsuccessful publishers who could not provide the books and the other publishers who have the same satisfaction index as initial. We have assumed that after applying retry on the worst situation, the unsatisfied customers will be satisfied as they are getting their desired books and the unsuccessful publishers who could not fulfill the orders properly will be satisfied because OBS did not cancel the orders which implies the good relationship level and low risk level between them.

For the customers who get books after applying retry, the availability is 1, relationship level is 1 and risk level is 0.5 as they are getting books (Table 4.10).

Properties	Level of importance	Level of property
Availability	1	1
Relationship level	0.5	1
Risk level	0.5	0.5

Table 4.10: Level of importance and properties after applying retry from satisfied customer's perspective

So, the satisfaction index of satisfied customer is:

$S_{C\text{-sat}} = 1 * 1 + 0.5 * 1 + 0.5 \div 0.5 = 2.5$

Satisfaction index of other customers remain same as initial because retry is not applied with them (Table 4.11).

Properties	Level of importance	Level of property
Availability	1	1
Relationship level	0.5	0
Risk level	0.5	1

Table 4.11: Level of importance and properties after applying retry from other customer's perspective

Satisfaction index of other customer which does not change is:

$S_{C\text{-others}} = 1 * 1 + 0.5 * 0 + 0.5 \div 1 = 1.5$

For the previously unsuccessful publishers who could not provide the required books, after applying retry action, the availability is 1, relationship level is 1 and risk level is 0.5 as they are providing books , so the relationship level will be increased and risk level will be decreased (Table 4.12).

Properties	Level of importance	Level of property
Availability	0.5	1
Relationship level	1	1
Risk level	0.5	0.5

Table 4.12: Level of importance and properties after applying retry from satisfied publisher's perspective

So, the satisfaction index of satisfied publisher is:

$S_{P\text{- sat}} = 0.5 * 1 + 1 * 1 + 0.5 \div 0.5 = 2.5$

The values of properties to compute satisfaction index of other publishers remain same as initial (Table 4.13).

Thus, the satisfaction index of other customers is:

$S_{P\text{-others}} = 0.5 * 1 + 1 * 0.5 + 0.5 \div 1 = 1.5$

Properties	Level of importance	Level of property
Availability	0.5	1
Relationship level	1	0.5
Risk level	0.5	1

Table 4.13: Level of importance and properties after applying retry from other publisher's perspective

The total satisfaction index of OBS after applying retry is:

$N_{C\text{-uns}} * S_{C\text{-sat}} + N_{C\text{-others}} * S_{C\text{-others}} + N_{P\text{-uns}} * S_{P\text{- sat}} + N_{P\text{-others}} * S_{P\text{-others}}$

Here, $N_{C\text{-uns}}$ = no. of unsatisfied customers,

$S_{C\text{-sat}}$ = satisfaction index of the satisfied customers, (unsatisfied customers are now satisfied after applying retry)

$N_{C\text{-others}}$ = no. of remaining customers,

$S_{C\text{-others}}$ = satisfaction index of remaining customers,

$N_{P\text{-uns}}$ = no. of unsuccessful publishers,

$S_{P\text{- sat}}$ = satisfaction index of the satisfied publishers, [unsuccessful publishers are now satisfied after applying retry]

$N_{P\text{-others}}$ = no. of remaining publishers,

$S_{P\text{-others}}$ = satisfaction index of remaining publishers.

4.3.1.4 Economic value after applying retry

As we are trying to compute the economic value of OBS separately for both retry and substitution, we have assumed that after applying retry action, customers get their desired books. As a result the satisfaction index of the customers and publishers about the OBS increases. Though the revenue remains same as before but this increasing satisfaction index has a significant effect on the revenue. In the value computation, this usually increases the total value of the OBS.

Value after retry, $V_{wr} = R_{wy} - C_{wry} + (Sat_r \div Sat_i) * R_{wy}$

Where, R_{wy} = yearly revenue of the OBS,

C_{wry} = yearly retry cost of the OBS,

Sat_r = satisfaction index after retry,

Sat_i = satisfaction index at the initial level.

4.3.2 Computation of value after applying Substitution

After applying substitution the expenditure of the OBS increases, revenue remains same but the satisfaction index changes. This change in satisfaction index affects the revenue and the value changes. Here we have assumed that customers get their desired books after substitution.

4.3.2.1 Cost after applying substitution

We have applied substitution when a publisher is not able to provide the book required by the customer. OBS sends a number of queries to different publishers asking the availability of the requested book. Thus, the cost for substitution per unsuccessful request is defined by the number of query per unsuccessful request and average transaction cost per query. This cost is then multiplied by the number of unsuccessful request per month and added with monthly cost of the OBS. In this way we get the monthly substitution cost. Then we compute the yearly substitution cost.

Substitution cost per unsuccessful request, $C_{qu} = N_{qpu} * C_t$

Substitution cost per month, $C_{ws} = C_w + N_u * C_{qu}$

Yearly substitution cost, $C_{wsy} = W_c + 12 * C_{ws}$

Where, N_{qpu} = number of query per unsuccessful request,

C_t = average transaction cost per each query,

C_w = Monthly cost of the OBS,

N_u= Number of unsuccessful request,

W_c= Yearly domain cost.

4.3.2.2 Revenue after applying substitution

In case of substitution also, the revenue remains same as before though after applying substitution, the changes in satisfaction of the adjacent partners add some value to the revenue of the OBS.

4.3.2.3 Satisfaction index after applying substitution

In the computation of satisfaction index after applying substitution, the customers are divided into two groups: a group of customers who are getting books after applying substitution and the remaining customers. As some publishers are substituted by some other publishers, the publishers are divided into three groups in substitution: one group who gets the orders, one group who are unable to provide the requested books and the other group who are not involved with any transaction of books. We have assumed that after applying substitution the unsuccessful publishers will be replaced by some other publishers. The satisfaction index of the publishers who are substituted will be less than before as the relationship level goes to low because of unavailability, thus the risk level becomes high. The satisfaction index of the publishers who get the orders is more than before as they are able to provide the requested books, thus the relationship level goes high and risk level goes low (Table 4.14).

Properties	Level of importance	Level of property
Availability	1	1
Relationship level	0.5	1
Risk level	0.5	0.5

Table 4.14: Level of importance and properties after applying substitution from satisfied customer's perspective

The satisfaction index of the customers who get the book after applying substitution will be same as the case in retry:

$S_{C\text{-}sat} = 1 * 1 + 0.5 * 1 + 0.5 \div 0.5 = 2.5$

The values of properties to compute satisfaction index of other customers remain same as initial (Table 4.15) .Thus, the satisfaction index of other customers is:

$S_{C\text{-}others} = 1 * 1 + 0.5 * 0 + 0.5 \div 1 = 1.5$

Properties	Level of importance	Level of property
Availability	1	1
Relationship level	0.5	0
Risk level	0.5	1

Table 4.15: Level of importance and properties after applying substitution from other customer's perspective

The publishers who are getting the orders after applying substitution become more satisfied about the OBS, so the availability is 1, relationship level is 1 and risk level is 0.5.As they are providing books, so the relationship level will be increased and risk level will be decreased (Table 4.16).

Properties	**Level of importance**	**Level of property**
Availability	0.5	1
Relationship level	1	1
Risk level	0.5	0.5

Table 4.16: Level of importance and properties after applying substitution from satisfied publisher's perspective

Satisfaction index of the publishers who get the order after applying substitution is:

$S_{P\text{- sat}} = 0.5 * 1 + 1 * 1 + 0.5 \div 0.5 = 2.5$

The publishers who have been substituted become unsatisfied with the OBS as they are losing the orders. Because of the unavailability of the requested book the relationship level goes low and risk level goes high (Table 4.17).

Properties	**Level of importance**	**Level of property**
Availability	0.5	0
Relationship level	1	0
Risk level	0.5	1

Table 4.17: Level of importance and properties after applying substitution from unsatisfied publisher's perspective

Satisfaction index of the publishers who are substituted is:

$S_{P\text{- uns}} = 0.5 * 0 + 1 * 0 + 0.5 \div 1 = 0.5$

Satisfaction index of other publishers remains same as initial (Table 4.18). Thus, we get:

$S_{P\text{-others}} = 0.5 * 1 + 1 * 0.5 + 0.5 \div 1 = 1.5$

Properties	**Level of importance**	**Level of property**
Availability	0.5	1
Relationship level	1	0.5
Risk level	0.5	1

Table 4.18: Level of importance and properties after applying substitution from other publisher's perspective

The total satisfaction index after applying substitution is:

$N_{C\text{-uns}} * S_{C\text{-sat}} + N_{C\text{-others}} * S_{C\text{-others}} + N_{P\text{-uns}} * S_{P\text{- uns}} + N_{P\text{-rep}} * S_{P\text{- sat}} + N_{P\text{-others}} * S_{P\text{-others}}$

Here, $N_{C\text{-uns}}$ = no. of unsatisfied customers,

$S_{C\text{-sat}}$ = satisfaction index of the satisfied customers,

$N_{C\text{-others}}$ = no. of remaining customers,

$S_{C\text{-others}}$ = satisfaction index of remaining customers,

$N_{P\text{-uns}}$ = no. of unsuccessful publishers,

$S_{P\text{- uns}}$ = satisfaction index of unsuccessful publishers,

$N_{P\text{-rep}}$ = no. of publishers who are replacement of the unsuccessful publishers,

$S_{P\text{- sat}}$ = satisfaction index of the satisfied publishers who get the order as replacement,

$N_{P\text{-others}}$ = no. of remaining publishers,

$S_{P\text{-others}}$ = satisfaction index of remaining publishers.

4.3.2.4 Economic value after applying Substitution

We have assumed that after applying substitution the satisfaction of the customers increases because anyway they are getting the book. The satisfaction of the publishers who are getting the orders increases whereas the satisfaction of the other publishers who are substituted decreases. These changes in the satisfaction index affect the revenue which is added with the profit.

Value after substitution, $V_{ws} = R_{wy} - C_{wsy} + (Sat_s \div Sat_i) * R_{wy}$

Where, R_{wy} = yearly revenue of the OBS,

C_{wsy} = yearly cost for substitution,

Sat_s = satisfaction index after substitution,

Sat_i = satisfaction index at the initial level.

4.4 Simulator

We have implemented a simulator which computes the economic value of an organization depending upon the input values provided by the user. This tool is built considering the users who are associated with e-businesses like Online Book Shop. It is very useful for the web based applications with web services. As our thesis mainly focuses on two repair actions i.e.

retry and substitution, this tool provides a comparative result among a worst case scenario, scenario after applying retry and substitution. Here worst case scenario is the case in which a number of web service failures are occurred and no repair action has been applied. Then retry and substitution are applied separately on the worst case scenario as web service repair actions. The simulator computes economic value of the organization in each of the three cases and provides a graphical comparison among them. The advantage of this simulator is that the users will be able to know which type of repair action should be applied in which situation. The tool is implemented using .NET Framework 3.5 and Visual C# 3.0. Fig 4.1 shows the Home page of the tool.

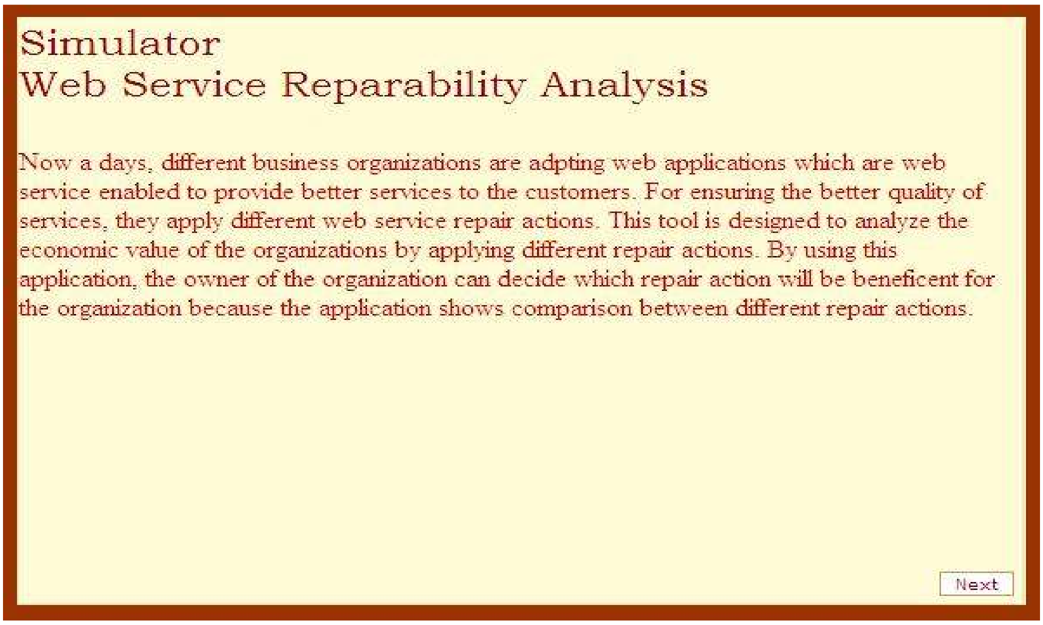
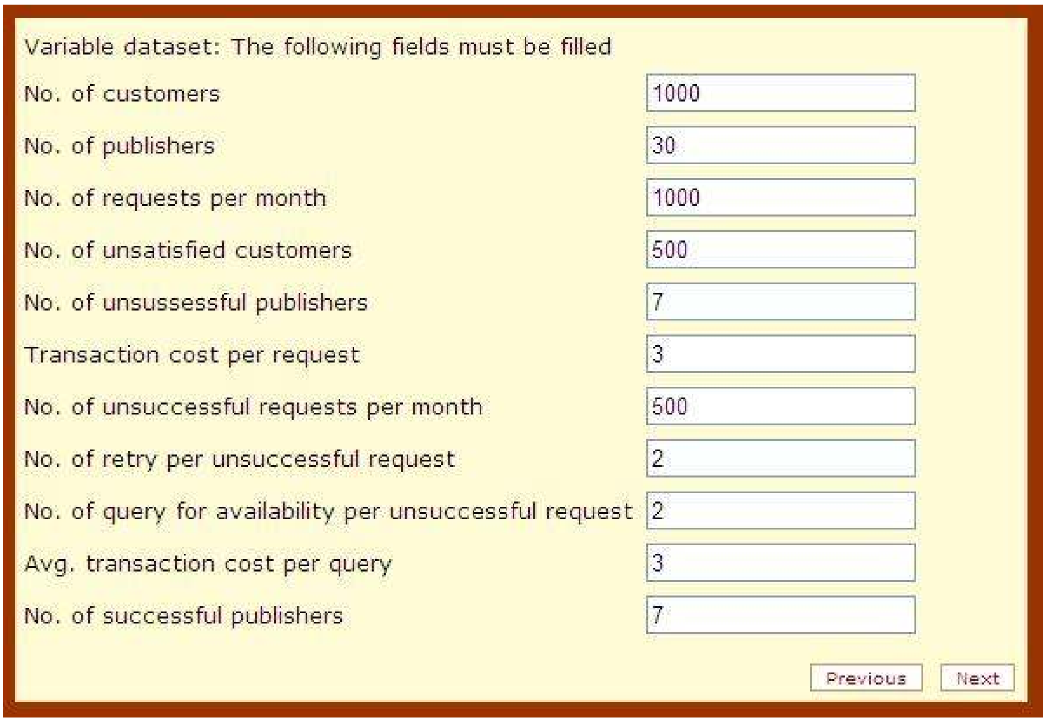


Fig 4.1: The Home page of the simulator for web service repairability analysis

In this simulator user provides some inputs that are necessary for the computation. The input set is divided in two groups. They are Variable dataset and Fixed dataset. Our analysis is mainly based on the variable data set in which user can alter the inputs to see the variation is results. Fixed dataset consists of all the fixed values of the organization which are taken directly to the computation. As they are important for the computation but they do not contribute to our analysis, we have considered this dataset as Fixed dataset from which we have taken all the fixed values of the organization For both datasets, there are some default values which can be changed by the user. Fig 4.1 and 4.2 are the snapshots of Variable dataset and Fixed dataset of the simulator.

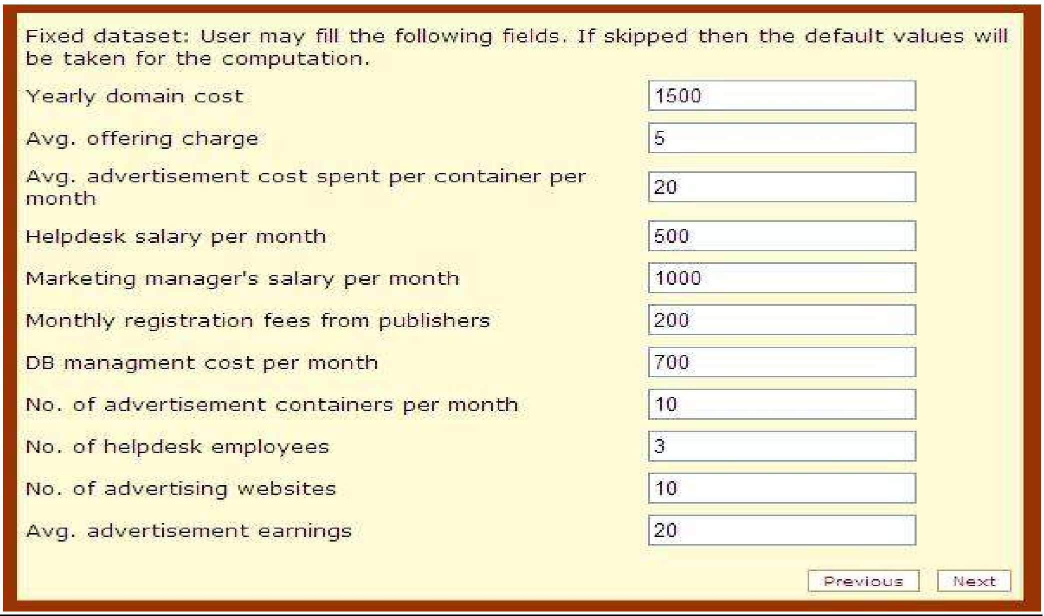


Variable dataset: The following fields must be filled

No. of customers	1000
No. of publishers	30
No. of requests per month	1000
No. of unsatisfied customers	500
No. of unsussessful publishers	7
Transaction cost per request	3
No. of unsuccessful requests per month	500
No. of retry per unsuccessful request	2
No. of query for availability per unsuccessful request	2
Avg. transaction cost per query	3
No. of successful publishers	7

Previous Next

Fig 4.2: Variable dataset. In this dataset, user can not keep any field empty



Fixed dataset: User may fill the following fields. If skipped then the default values will be taken for the computation.

Yearly domain cost	1500
Avg. offering charge	5
Avg. advertisement cost spent per container per month	20
Helpdesk salary per month	500
Marketing manager's salary per month	1000
Monthly registration fees from publishers	200
DB managment cost per month	700
No. of advertisement containers per month	10
No. of helpdesk employees	3
No. of advertising websites	10
Avg. advertisement earnings	20

Previous Next

Fig 4.3: Fixed dataset. If user keeps any field empty in this dataset, the default value will be taken for the computation

The inputs provided by the user are kept in an XML file which can be accessed from any other software and field. Fig 4.3 is the snapshot of XML file. We used **`XmlTextWriter`** class for writing the XML file. From this XML file, all the inputs are taken for the computation of economic values in different scenarios. After all the computations, the results are shown.

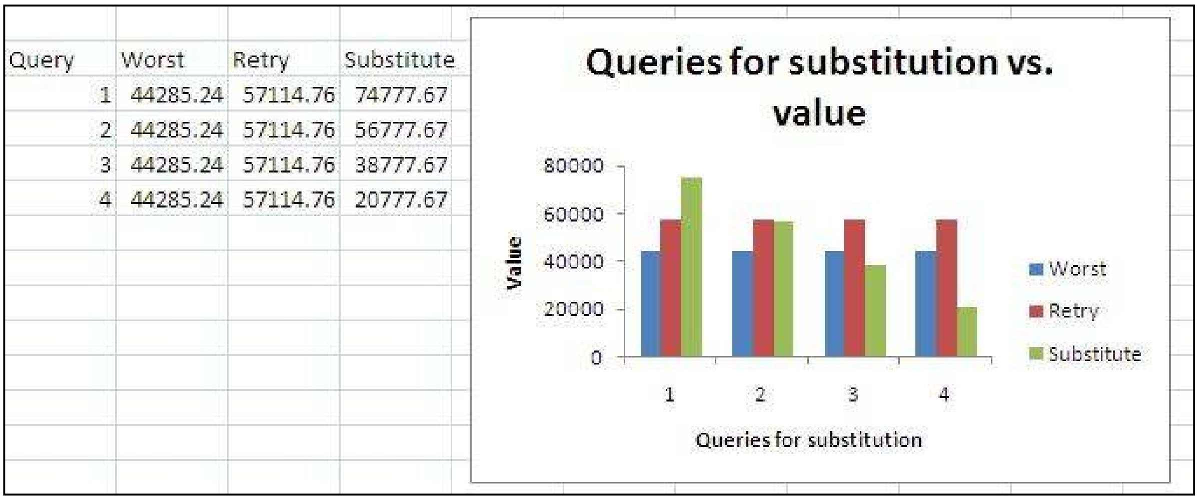

```xml
<?xml version="1.0" encoding="utf-8" ?>
- <Organization>
  - <Variable_values>
      <No_of_customers>1000</No_of_customers>
      <No_of_publishers>30</No_of_publishers>
      <No_of_requests_per_month>1000</No_of_requests_per_month>
      <No_of_unsatisfied_customers>500</No_of_unsatisfied_customers>
      <No_of_unsuccessful_publishers>7</No_of_unsuccessful_publishers>
      <Transaction_cost_per_request>3</Transaction_cost_per_request>
      <No_of_unsuccessful_requests_per_month>500</No_of_unsuccessful_requests_per_month>
      <No_of_retry_per_unsuccessful_request>2</No_of_retry_per_unsuccessful_request>
      <No_of_query_for_availability_per_unsuccessful_request>2</No_of_query_for_availability_per_unsuccessful_request>
      <Avg_transaction_cost_per_query>3</Avg_transaction_cost_per_query>
      <No_of_successful_publishers>7</No_of_successful_publishers>
    </Variable_values>
  - <Fixed_values>
      <Yearly_domain_cost>1500</Yearly_domain_cost>
      <Avg_offering_charge>5</Avg_offering_charge>
      <Avg_advertisement_cost_spent_per_container_per_month>20</Avg_advertisement_cost_spent_per_container_per_month>
      <Helpdesk_salary_per_month>500</Helpdesk_salary_per_month>
      <Marketing_managers_salary_per_month>1000</Marketing_managers_salary_per_month>
      <Monthly_registration_fees_from_publishers>200</Monthly_registration_fees_from_publishers>
      <DB_management_cost_per_month>700</DB_management_cost_per_month>
      <No_of_advertisement_containers_per_month>10</No_of_advertisement_containers_per_month>
      <No_of_helpdesk_employees>3</No_of_helpdesk_employees>
      <No_of_advertising_websites>10</No_of_advertising_websites>
      <Avg_advertisement_earnings>20</Avg_advertisement_earnings>
    </Fixed_values>
  </Organization>
```


Fig 4.4: XML file generated after providing the inputs

We used Microsoft Excel Charts to create a graphical representation of the result. Here we used Column chart. There are three columns in each comparison where each column represents a specific scenario. In the implementation, we used `Workbook`, `Worksheet`, `ChartObjects` and `Chart` classes to create this graphical representation.

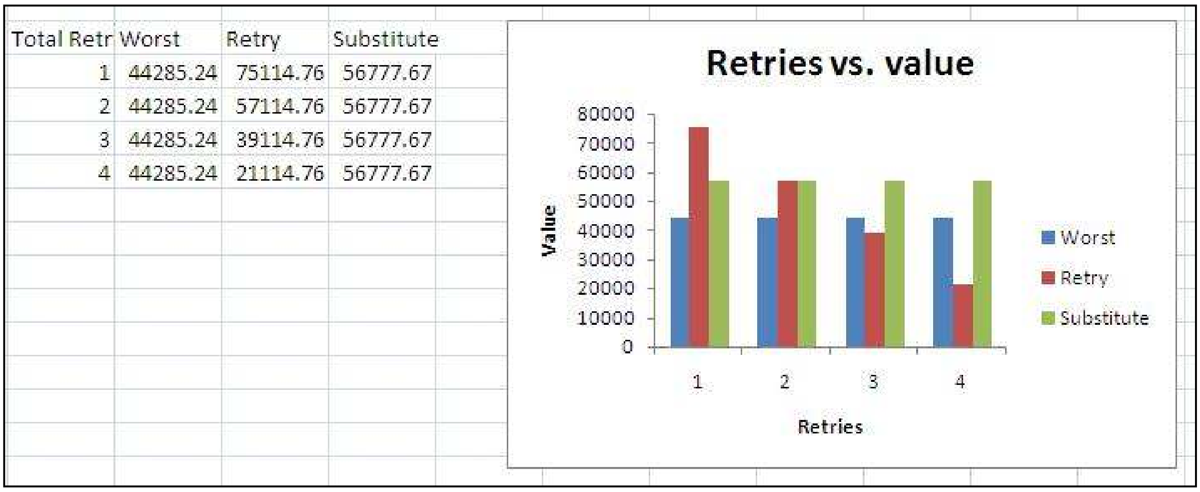
```
Workbook wb = xla.Workbooks.Add(XlSheetType.xlWorksheet);
Worksheet ws = (Worksheet)xla.ActiveSheet;
// Now create the chart.
ChartObjects chartObjs =
   (ChartObjects)ws.ChartObjects(Type.Missing);
//for the graph of no. of cust and value
ChartObject chartObj = chartObjs.Add(200, 20, 300, 200);
Chart xlChart = chartObj.Chart;
```

The simulator provides seven different graphs representing seven different types of analysis in our thesis. Fig 4.5 and 4.6 are some of the snapshots of the result.



Query	Worst	Retry	Substitute
1	44285.24	57114.76	74777.67
2	44285.24	57114.76	56777.67
3	44285.24	57114.76	38777.67
4	44285.24	57114.76	20777.67

Fig 4.5: Graphical representation of number of queries for substitution vs. value



Total Retr	Worst	Retry	Substitute
1	44285.24	75114.76	56777.67
2	44285.24	57114.76	56777.67
3	44285.24	39114.76	56777.67
4	44285.24	21114.76	56777.67

Fig 4.6: Graphical representation of number of retries vs. value

This simulator is a comparative study of web service repairability. Our intention was to build a tool which can give us different results depending on user input. From these results we can get an abstract idea about the affects of repair actions on the economic value of organization. We have found some significant results from this analysis which will be discussed in Chapter 5. The future extension of this tool can be to extend this simulator as a web service so that different web applications can use it at a time of applying any repair action considering the affects. During the implementation of this simulator, our attention was devoted to only two repair actions i.e. retry and substitution as they are the main concern of our thesis. But applying other repair actions like compensation, redo can also be considered here. This can be another extension of our tool.

CHAPTER 5

Result and comparison

Our analysis is aiming to apply repair strategies on the web service of our proposed model and to observe how the repair strategies can help to recover from the worst situation. To accomplish this, we have computed economic values considering three different cases: value in worst case, value after applying retry and value after applying substitution. We have assumed a data set consisting of reasonable values and we have changed the values of this data set to find out the changes in economic values after applying proposed repair strategies. In the following discussion we show the result of our work and the comparison between different graphs.

5.1 Date set

For ease of our analysis, we have used a reasonable data set to compute different equations of our model. We have used two data sets: variable data set and fixed data set. Variable data have been changed to get some significant results and comparisons (Table 5.1) whereas fixed data set remain same as given in Table 5.2 as changes in these data are not producing any significant results. We have assumed some values for this data set which is quite reasonable for our proposed model. The data sets are given in the following:

Parameter	Value
Number of customers per month	1000
Number of unsatisfied customers per month	500
Number of publishers per month	30
Number of unsuccessful publishers per month	7
N Number of successful publishers per month	7
Number of requests per month	1000
Number of unsuccessful requests per month	500
Average transaction cost per try	3
Average transaction cost per query for availability	3
Number of retry per unsuccessful request	2
Number of query for availability	2

Table 5.1: Variable Data Set

Parameter	Value
Yearly domain cost	1500
Avg. offering charge	5
Avg. advertisement cost spent per container per month	20
Helpdesk salary per month	500
Marketing managers salary per month	1000
Monthly registration fees from publishers	200
DB management cost per month	700
Number of advertisement containers per month	10
Number of helpdesk employees	3
Number of advertising websites	10
Average advertisement earnings	20

Table 5.2: Fixed data Set

5.2 Number of retry vs. value

To recover from the worst case situation, it is necessary to know how many times the retry actions should be taken for each unsuccessful request. We have assumed that the retry actions will be taken until the value is better than the worst case. Whenever the value decreases than the worst case, any more retry action should not be taken as it is worthless to increase the cost without having better situation. With the given data set (Section 5.1), we have found the following results:

No. of Retry	Value in Worst case	Value after applying Retry	Value after applying Substitution
1	44285.24	75114.75	56777.67
2	44285.24	57114.76	56777.67
3	44285.24	39114.76	56777.67
4	44285.24	21114.76	56777.67

Table 5.3: Data for number of retry vs. value

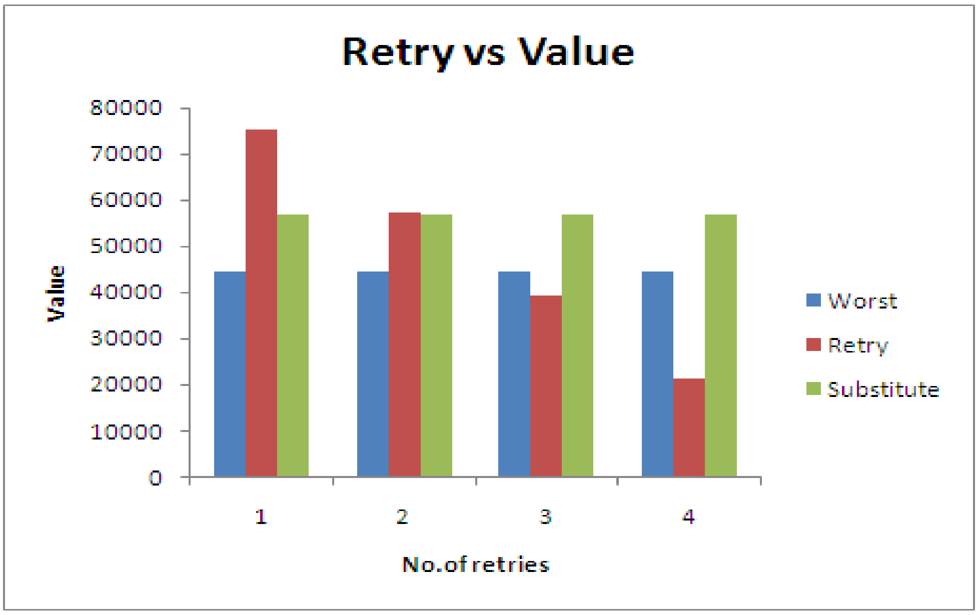


Fig 5.1: Graph of retry vs. value. It shows how many retries should be applied as repair action

From Fig 5.1, it can be observed that the value of retry decreases as the number of retry increases. As each retry adds cost to the usual cost of OBS, thus the value decreases with each additional retry. This graph also shows that up to two retry actions the value of retry remains better than the worst case and the value of substitution. At the third retry, value of retry decreases than the worst case and the value of substitution which indicates that OBS should not apply more than two retry actions because it will add more costs but will not provide any better situation.

5.3 Number of query for availability vs. value

When we apply substitution as a repair action to any unsuccessful request we follow a number of queries to the publishers to ensure that the required book is available to provide the customer. We have assumed that the publisher who responses first will be chosen to proceed the order. If there are more publishers responding at the same time then OBS will ask Customer to choose among them. For our analysis, it is interesting to find out how many queries are necessary for applying substitution. With the given data set (Section 5.1), the result is shown in the following:

No. of query for availability	Value in Worst case	Value after applying Retry	Value after applying Substitution
1	44285.24	57114.76	74777.67
2	44285.24	57114.76	56777.67
3	44285.24	57114.76	38777.67
4	44285.24	57114.76	20777.67

Table 5.4: Data for number of query for availability vs. value

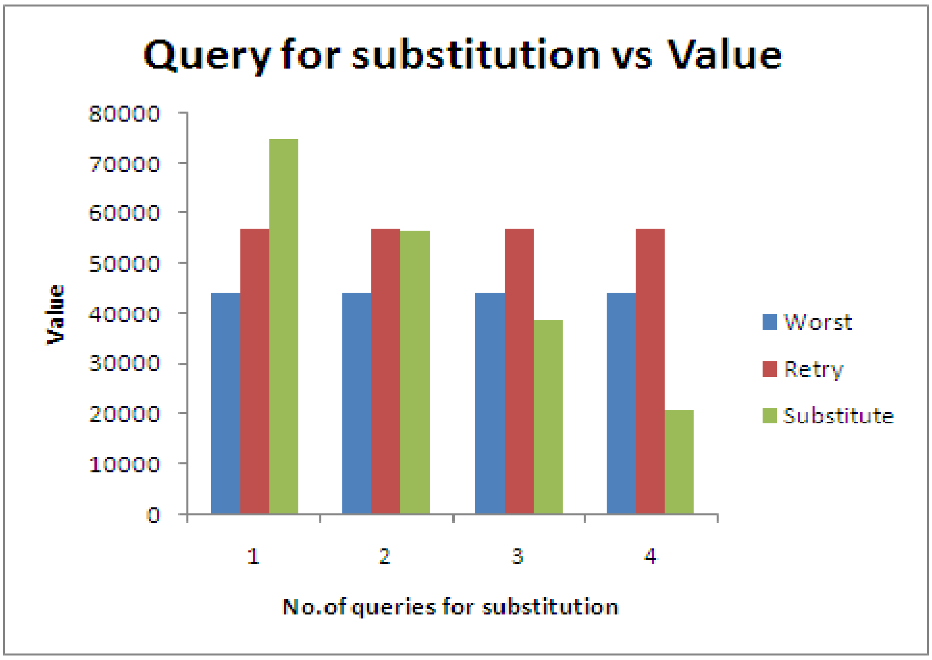


Fig 5.2: Graph of query vs. value. It shows how many queries should be applied for substitution

In Fig. 5.2, the value of substitution decreases as the number of query for availability increases because each query adds some additional costs. Up to two queries the value of substitution remains better than the worst case but at the second query it becomes less the value of retry. If three queries are applied for getting the availability information of the particular book then the value of substitution becomes less than the worst case which indicates that OBS should query to two publishers at a time for availability information and not more than that in the given data set (Section 5.1).

5.4 Number of unsuccessful requests vs. value

In our analysis, we are dealing with unsuccessful requests on which the repair actions are applied. Unsuccessful requests are considered as requests which have failed to complete the whole process. Considering the given data set (Section 5.1), we have tried to capture the changes with increasing number of unsuccessful requests which is given in the following:

No. of unsuccessful requests	Value in Worst case	Value after applying Retry	Value after applying Substitution
200	44285.24	78714.76	78377.67
250	44285.24	75114.76	74777.67
300	44285.24	71514.76	71177.67
350	44285.24	67914.76	67577.67
400	44285.24	64314.76	63977.67
450	44285.24	60714.76	60377.67
500	44285.24	57114.76	56777.67
550	44285.24	53514.76	53177.67
600	44285.24	49914.76	49577.67
650	44285.24	46314.76	45977.67
700	44285.24	42714.76	42377.67
750	44285.24	39114.76	38777.67

Table 5.5: Data for number of unsuccessful requests vs. value

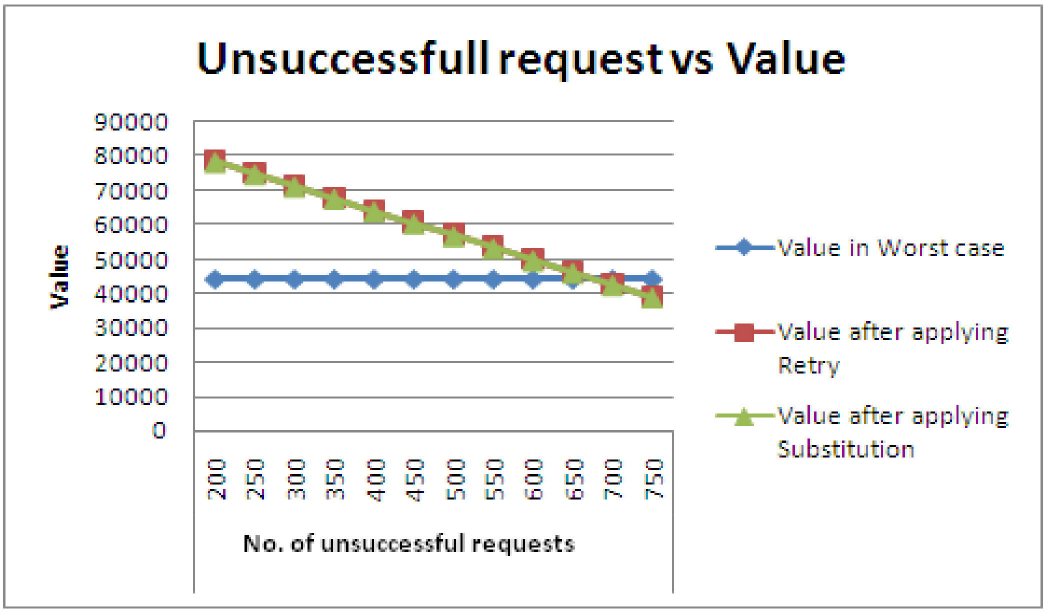


Fig 5.3: Graph of unsuccessful requests vs. value. It shows the changes in value with the increasing number of unsuccessful requests after applying retry and substitution

From Fig 5.3, it can be observed that as the number of unsuccessful requests increases the value after applying retry and substitution decreases. Each unsuccessful request requires a repair action to recover from the worst situation which adds more costs. For these additional costs both of the values decrease and become less than the worst case after some limit. In the given data set (Section 5.1), the value becomes less than the worst case when the number of unsuccessful requests is 700 within 1000 requests.

5.5 Number of query vs. value depending on query cost

The cost is an important factor to consider while applying repair actions. It is worthless to apply repair actions which cannot repair the unsuccessful requests but increase expenditure. Variation in query cost can change the number of queries for availability information to apply. It is necessary to get the information of the availability of the requested book for applying substitution but the cost should be reasonable too. It is obvious that as less the query cost, more queries can be applied. But how many queries can be applied in the given data set (Section 5.1) with different costs can be an interesting observation. To observe that, we have captured the graphs for query cost 1, 2, 3 and 4.

Query cost 1			
No. of Query	**Value in Worst Case**	**Value after applying Retry**	**Value after applying Substitution**
1	44285.24	57114.76	86777.67
2	44285.24	57114.76	80777.67
3	44285.24	57114.76	74777.67
4	44285.24	57114.76	20777.67
Query cost 2			
No. of Query	**Value in Worst Case**	**Value after applying Retry**	**Value after applying Substitution**
1	44285.24	57114.76	80777.67
2	44285.24	57114.76	68777.67
3	44285.24	57114.76	56777.67
4	44285.24	57114.76	20777.67
Query cost 3			
No. of Query	**Value in Worst Case**	**Value after applying Retry**	**Value after applying Substitution**
1	44285.24	57114.76	74777.67
2	44285.24	57114.76	56777.67
3	44285.24	57114.76	38777.67
4	44285.24	57114.76	20777.67
Query cost 4			
No. of Query	**Value in Worst Case**	**Value after applying Retry**	**Value after applying Substitution**
1	44285.24	57114.76	68777.67
2	44285.24	57114.76	44777.67
3	44285.24	39114.76	20777.67
4	44285.24	21114.76	20777.67

Table 5 6: Data for number of query vs. value depending on the query cost

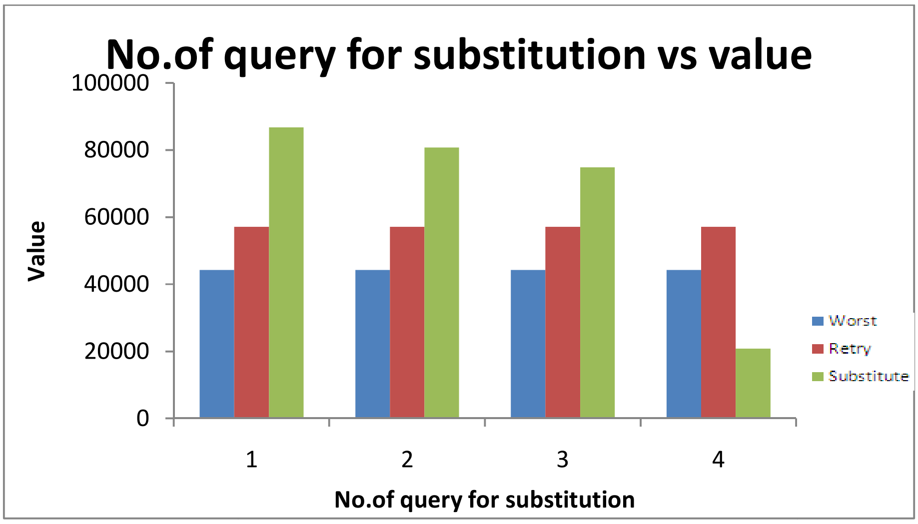


Fig 5.4: Graph of query vs. value. (when query cost is 1)

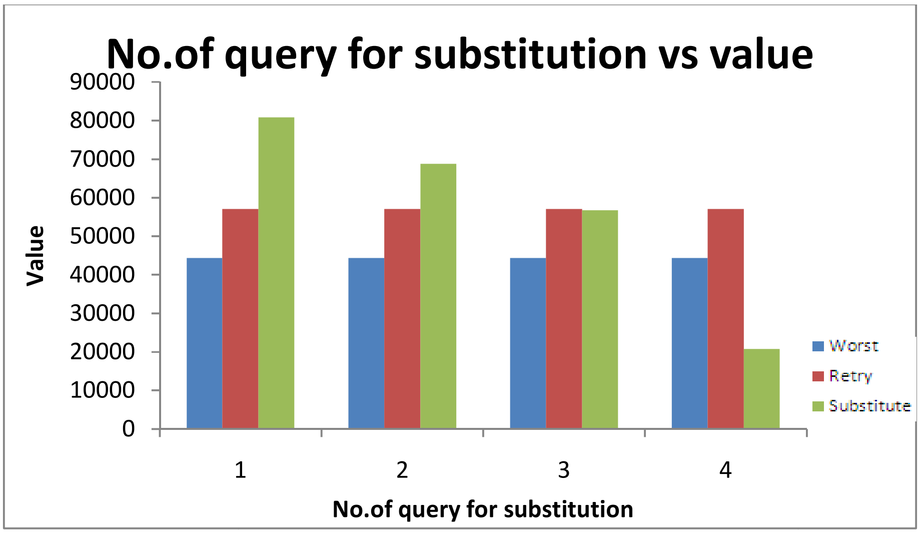


Fig 5.5: Graph of query vs. value (when query cost is 2)

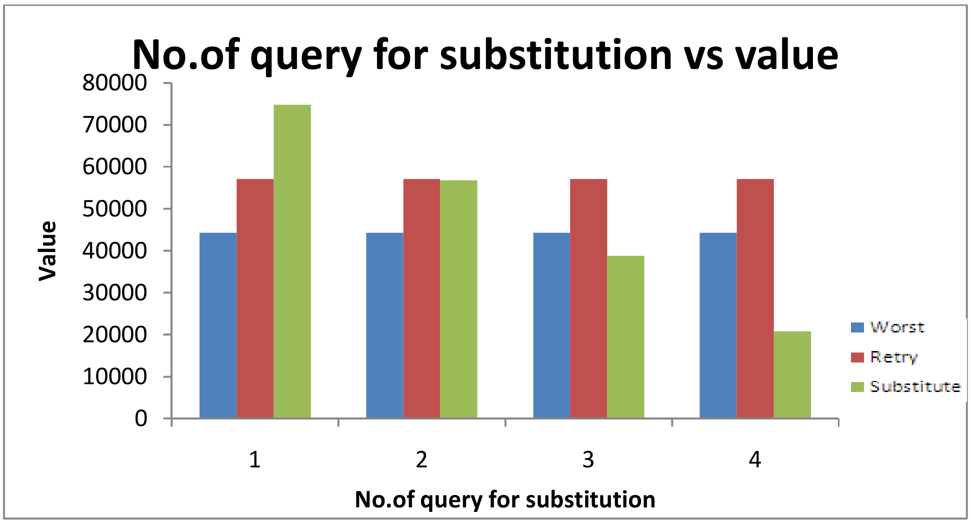


Fig 5.6: Graph of query vs. value (when query cost is 3)

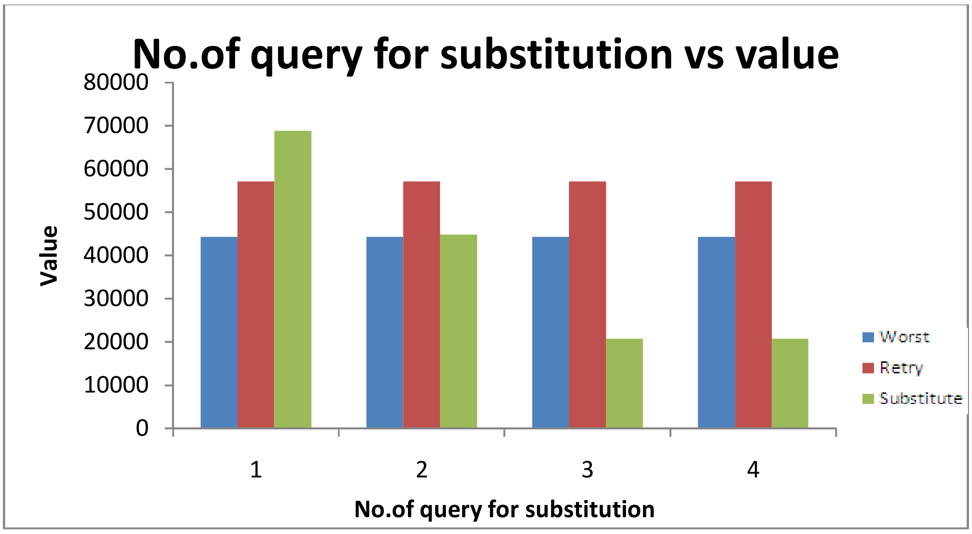


Fig 5.7: Graph of query vs. value. (when query cost is 4)

From Fig 5.4, 5.5, 5.6 and 5.7 it can be easily observed that as the cost of the query increases the value after applying substitution decreases. For query cost 1 and 2, the value after applying substitution is better than the value in worst case up to three queries but decreases than the worst case after applying more queries. For query cost 3 and 4 two queries are possible to make the situation better than the worst case. Another observation from these graphs is when the value after applying substitution becomes less than the value after applying retry. For query cost 1 and 2, the value after applying substitution becomes less than value after applying retry at fourth and third query respectively. On the contrary, the value after applying substitution decreases than the value after applying retry at the second query when query cost is 3 and 4.These graphs help the OBS to decide how many queries should be applied to get the availability information by keeping the query cost reasonable.

5.6 Number of retry vs. value depending on transaction cost

In our analysis, transaction cost of a request includes the cost of processing and fulfilling the order. When a failure occurs and a request is not fulfilled then retry or substitution is applied. Each retry adds additional transaction cost to the expenditure of OBS. With increasing transaction cost the total cost increases which results decrease in value. It is interesting to know how the value changes with different transaction costs. To observe that we have used the given data set (Section 5.1) with transaction cost 1, 2 and 3.

In Fig 5.8, 5.9 and 5.10, it is shown that as the transaction cost increases the value after applying retry decreases with the increasing number of retries. With transaction cost 1, the value after applying retry becomes less than the value in worst case if more than eight retries are applied. For transaction cost 2 and 3, up to four and two retries are possible respectively to have better situation than worst case. From these graphs it is also observed that at transaction cost 1, the value of retry becomes less than the value of substitution at the seventh retry whereas with transaction cost 2, value of retry becomes less than the value of substitution at fourth retry. On the contrary, the value of retry becomes less than the value of substitution at the third retry when transaction cost is 3. From these graphs, the OBS can decide how many retries it can apply for each unsuccessful request by keeping the transaction cost reasonable.

Transcation Cost 1			
No. of Retry	**Value in Worst Case**	**Value after applying Retry**	**Value after applying Substitution**
6	68285.24	81114.76	80777.67
7	68285.24	75114.76	80777.67
8	68285.24	69114.76	80777.67
9	68285.24	63114.76	80777.67
Transcation Cost 2			
No. of Retry	**Value in Worst Case**	**Value after applying Retry**	**Value after applying Substitution**
2	56285.24	81114.76	68777.67
3	56285.24	69114.76	68777.67
4	56285.24	57114.76	68777.67
5	56285.24	45114.76	68777.67
Transcation Cost 3			
No. of Retry	**Value in Worst Case**	**Value after applying Retry**	**Value after applying Substitution**
1	44285.24	75114.76	56777.67
2	44285.24	57114.76	56777.67
3	44285.24	39114.76	56777.67
4	44285.24	21114.76	56777.67

Table 5 7: Data for Number of retry vs. value depending upon the transaction cost

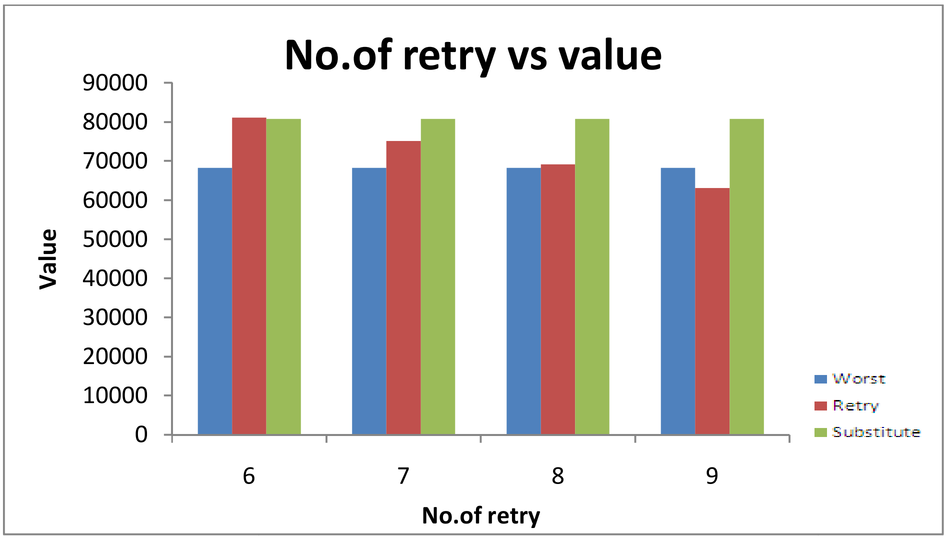


Fig 5.8: Graph of retry vs. value (when transaction cost is 1)

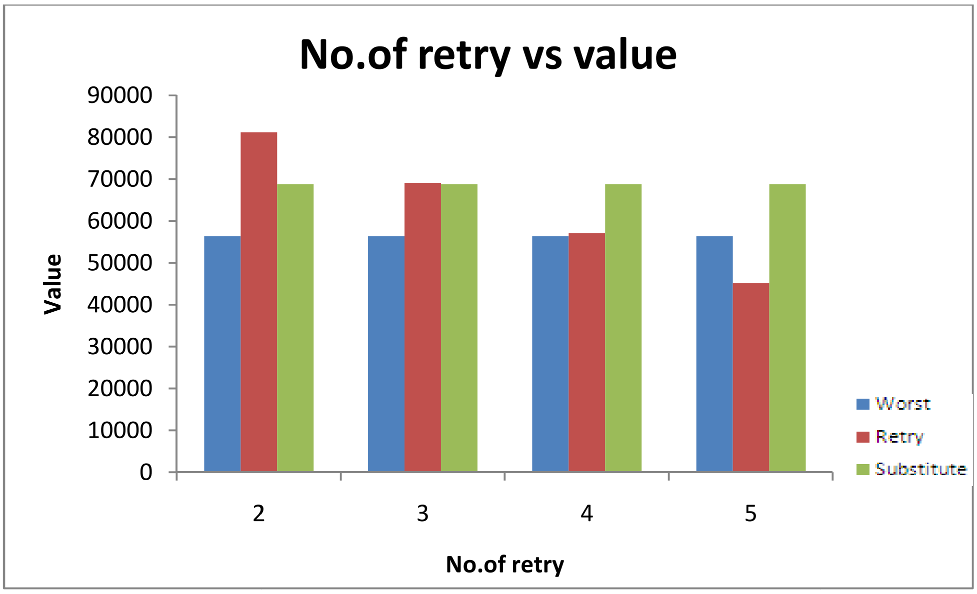


Fig 5 9: Graph of retry vs. value (when transaction cost is 2)

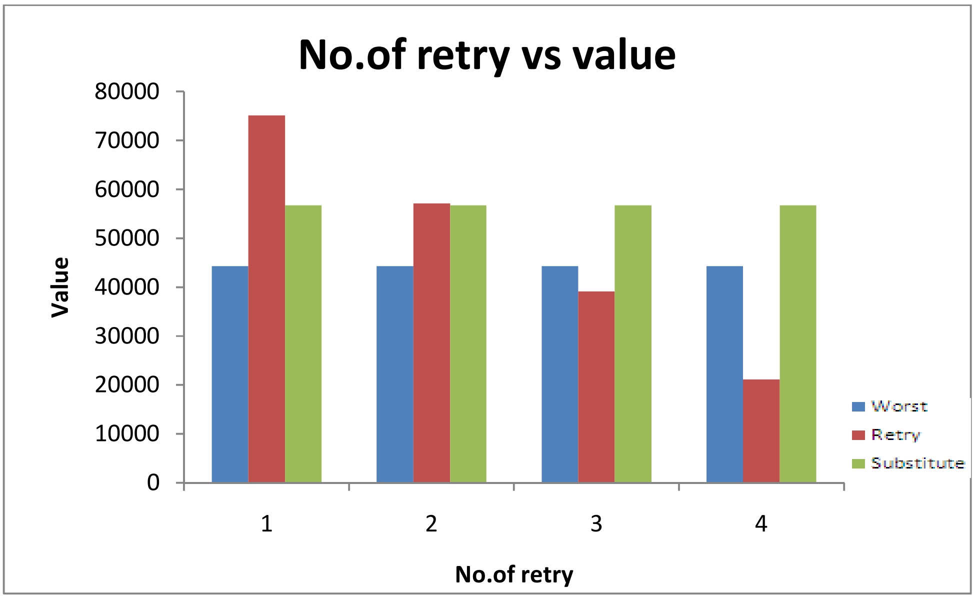


Fig 5.10: Graph of retry vs. value (when transaction cost is 3)

CHAPTER 6

Conclusion

Self-repairability of Web services is necessary to get rid of service disruptions which result in low satisfaction index, loss of customers, and a multitude of financial repercussions. Web services that adapt to the continuous changes in the environment are considered as self-healable. Self-healing Web services means embodying them with extra functionalities, which permit them to handle operation resumption after crashes. Hard-wiring these functionalities into the code of a Web service results in a Web service that is difficult to maintain. For easy maintenance, any Web service should have self-repairability and for this reason, in our analysis retry and substitution are applied as repair strategies to make our proposed e-business scenario self-healable.

6.1 Concluding Remarks

The aim of our analysis is to apply repair strategies in the Web service of an Online Book Shop (OBS) represented in our proposed model and observe how the economic value changes after applying these repair strategies. To achieve this goal, we have assumed a worst case scenario where number of unsuccessful requests is high and in that worst situation we have applied two repair strategies: Retry and Substitution to repair unsuccessful requests. When a failure occurs in the process execution, retry action repeats the whole process which includes additional transaction cost for each retry whereas for each substitution the OBS query to a number of publishers which includes additional query cost for each query. Our analysis contains three different computations: at worst case, after applying retry and after applying substitution. We have computed the expenditure, revenue, satisfaction index and economic value of OBS in these three cases. The aim of the OBS in our proposed e-business scenario is obviously to increase its economic value which includes both profit and satisfaction of customers. Satisfaction of adjacent partners is important to analyze as their high satisfaction can increase sales whereas low satisfaction can be the reason for loss in business. In our analysis, satisfaction index is computed to evaluate the satisfaction of customers and publishers about the OBS. We have considered some properties such as availability, relationship level and risk level to compute satisfaction index and set some importance level for each of these properties. This satisfaction index is added with the profit to get the economic value of OBS.

To accomplish the computation automatically we have implemented a simulator which takes input and generates outputs in graphs where each graph shows three different bars for value in worst case, value after applying retry and value after applying substitution.

By applying these repair actions we have found some results which is interesting to analyze. We have assumed that for any unsuccessful request any number of retries or queries for substitution can be applied. For ease of our analysis, we have considered a reasonable data set (Section 5.1) and observed how the economic value changes after applying retry and substitution compared to worst case scenario. For our analysis, sometimes we have changed the variable data set but keep the fixed data set as given and captured different graphs which generates different observations. With the given data set we have found that the economic value of retry becomes better than the worst case up to 2 retries which indicates that at most 2 retries can be applied to have better situation than the worst case. The economic value of substitution becomes better than the worst case up to 2 queries in the given data set which means that the OBS can query at most 2 publishers for each substitution for availability information of the particular book . Another interesting observation is that with repair strategies the OBS can handle up to 650 unsuccessful requests within 1000 requests which means that the economic values after applying retry and substitution remain better than the worst case up to 650 unsuccessful requests in the given data set. With more unsuccessful requests both of the values become less than the value in worst case. The results also show that number of retries and queries can be increased by reducing transaction cost and query cost respectively.

6.2 Future work

Through our work we have got some ideas about how the repair strategies can make a system more reliable to its adjacent partners. Our main concern is to apply repair strategies in the system of OBS by keeping the cost reasonable and to get a high economic value. In future, repair strategies can be applied in the system of publishers also. We have computed the economic values of OBS but the economic values of publishers could be done also.

In our analysis we have considered only two repair strategies but other repair strategies like compensation, redo, architectural reconfiguration could be applied in future to repair unsuccessful requests. Also Combination of repair strategies like two retries-one substitution, one retry-two substitutions, three retries-one substitution etc. could be applied for each unsuccessful request. In our work, before applying substitution the OBS query to a number of publishers to get the availability information of the particular book and whenever it gets a positive response from a publisher it replaced the previous unsuccessful publisher with this publisher. But in future the substitution could be done directly without querying to the publishers by depending on some factors like relationship level, risk level etc. or through some priority criteria.

Our analysis includes three properties of satisfaction index: availability, relationship level and risk level but in future it can be done by considering other properties. We have considered three importance levels: 1, 0.5 and 0 which could be different. Also we have assumed the values of availability, relation level and risk level as 1, 0.5 and 0 but we envision that it should be more fractional.

BIBLIOGRAPHY

1. *Web Services Architecture*, *W3C Working Group Note 11 February 2004*. [Online] http://www.w3.org/TR/ws-arch/#whatis

2. **Stephen Potts, Mike Kopack.** *Sams Teach Yourself Web Services in 24 Hours*. Sams, May 12, 2003. pp. 10-11

3. **Jim Conallen**. *Building Web Applications with UML*, 2nd Edition. Addison-Wesley Professional, October 13, 2002

4. *SOAP Tutorial*. [Online] http://www.w3schools.com/soap/default.asp

5. *Investigating the Potential of Service Oriented Architectures to Realize Dynamic Capabilities.* **Haresh Luthria, Fethi Rabhi, Michael Briers.** 2007 IEEE Asia-Pacific Services Computing Conference (APSCC 2007); Tsukuba Science City, Japan; December 2007. Proceedings of the 2nd IEEE Asia-Pacific Service Computing Conference; **IEEE**, Piscataway, USA, 2007; pp.390

6. *Characterizing and checking self-healability.* **Marie-Odile Cordier, Yannick Pencol´, Louise Trav´e-Massuy`and Thierry Vidal.** 2008. Proceeding of the 2008 conference on ECAI 2008: 18th European Conference on Artificial Intelligence. Pages 789-790.

7. *e-Sevice adaptation.* **Barbara Pernici and Pierluigi Plebani.** Publication on progress.

8. *Quality-Aware Design of Repairable Processes*. **Cappiello, C., Pernici, B.** In Proceedings of the 13th International Conference on Information Quality (ICIQ '08), MIT Press, Boston, MA, USA, November 2008

9. *WS-DIAMOND: Web Services-DIAgnosability, MONitoring and Diagnosis*. **WS-Diamond Team**. MIT press, 2009, pp. 213–239

10. *Estimating value in service systems: A case study of a repair service system.* IBM Systems. **Caswell, N. S. , Nikolaou, C. , Sairamesh, J. , Bitsaki, M. , Koutras, G. D. , Iacovidis, G.** Journal, Vol. 47, No. 1. (2008), pp. 87-100