Software Requirements Specification

For

HDFC- BANKING SYSTEM

Table of Contents

1. Introduction

The Housing Development Finance Corporation Limited (HDFC) was amongst the first to receive an 'in principle' approval from the Reserve Bank of India (RBI) to set up a bank in the private sector, as part of the RBI's liberalization of the Indian Banking Industry in 1994. The bank was incorporated in August 1994 in the name of 'HDFC Bank Limited', with its registered office in Mumbai, India. HDFC Bank commenced operations as a Scheduled Commercial Bank in January 1995.

This document gives detailed functional and nonfunctional requirements for the HDFC-bank management system. This product will support online banking transaction. The purpose of this document is that the requirements mentioned in it should be utilized by software developer to implement the system.

1.1 Purpose

HDFC-Online banking system provides is specifically developed for internet banking for Balance Enquiry, Funds Transfer to another account in the same bank, Request for cheque book/change of address/stop payment of cheques, Mini statements (Viewing Monthly and annual statements).

The Traditional way of maintaining details of a user in a bank was to enter the details and record them. Every time the user need to perform some transactions he has to go to bank and perform the necessary actions, which may not be so feasible all the time. It may be a hard-hitting task for the users and the bankers too. The project gives real life understanding of Internet banking and activities performed by various roles in the supply chain. Here, we(HDFC) provide an automation for banking system through Internet. Internet banking system project captures

activities performed by different roles in real life banking which provides enhanced techniques for maintaining the required in- formation up-to-date, which results in efficiency. The project gives real life understanding of Internet banking and activities performed by various roles in the supply chain.

1.2 Scope

This Product will automate of banking transaction process. This Project investigates the entry threshold for providing a new transaction service channel via the real options approach, where the entry threshold is established by using an Internet banking system designed for the use of normal users(individuals), Industrialists, Entrepreneurs, Educational Institutions(Financial sections), Organizations and Academicians under transaction rate uncertainty.

1.3 Overview

The system provides easy solution to banks.

Overview: The SRS will include two sections, namely:

Overall Description: This section will describe major components of the system, interconnections, and external interfaces.

Specific Requirements: This section will describe the functions of actors, their roles in the system and the constraints faced by sys- tem.

2. General description

2.1 Product Perspective:

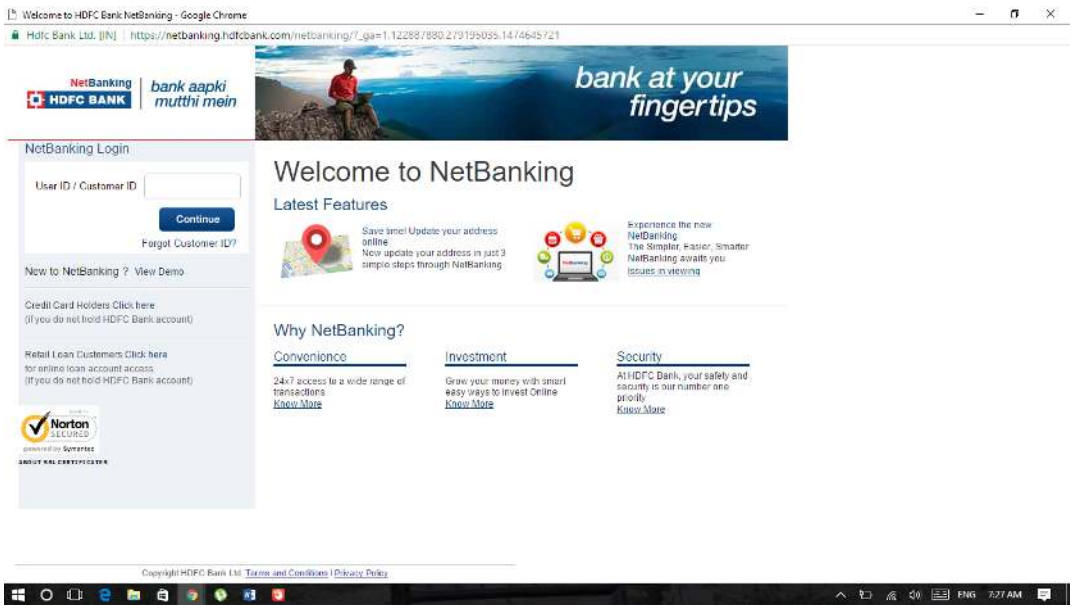
The client will have client interface in which he can interact with the banking sys tem. It is a web based interface which will be the web page of the banking application. Starting a page is displayed asking the type of customer he is whether ordinary or a corporate customer. Then the page is redirected to login page where the user can enter the login details. If the login particulars are valid then the user is taken to a home page where he has the entire transaction list that he can perform with the bank. All the above activities come under the client interface.

The administrator will have an administrative interface which is a GUI so that he can view the entire system. He will also have a login page where he can enter the login particulars so that he can perform all his actions. This administrative interface provides different environment such that he can maintain data- base & provide backups for the information in the database. He can register the users by providing them with username, password & by creating account in the database. He can view the cheque book request & perform action to issue the cheque books to the clients.

2.2 Software Interface:

Front End Client:

The system is a web based application clients are requiring using modern web browser such as Mozilla Firefox 1.5, PHP,Google Chrome.



Welcome to HDFC Bank NetBanking - Google Chrome

Hdfc Bank Ltd. [IN] | https://netbanking.hdfcbank.com/netbanking/?_ga=1.122887880.279195035.1474645721

NetBanking HDFC BANK | bank aapki mutthi mein

bank at your fingertips

NetBanking Login

User ID / Customer ID

Continue

Forgot Customer ID?

New to NetBanking ? View Demo

Credit Card Holders Click here
(If you do not hold HDFC Bank account)

Retail Loan Customers Click here
for online loan account access
(If you do not hold HDFC Bank account)

Welcome to NetBanking

Latest Features

Save time! Update your address online
Now update your address in just 3 simple steps through NetBanking

Experience the new NetBanking
The Simpler, Easier, Smarter NetBanking awaits you
Issues in viewing

Why NetBanking?

Convenience
24x7 access to a wide range of transactions
Know More

Investment
Grow your money with smart easy ways to invest Online
Know More

Security
At HDFC Bank, your safety and security is our number one priority
Know More

Copyright HDFC Bank Ltd. Terms and Conditions | Privacy Policy

*** Web Server:**

The web application will be hosted on one of the apache server.

*** Back End:**

We use backend as MY SQL.

3. Functional Specifications

This section provides the functional overview of the product. The project will require the PHP as a front end and at the back end the database MYSQL will be running. Various functional modules that can be implemented by the product will be

1. Login

2. Validation

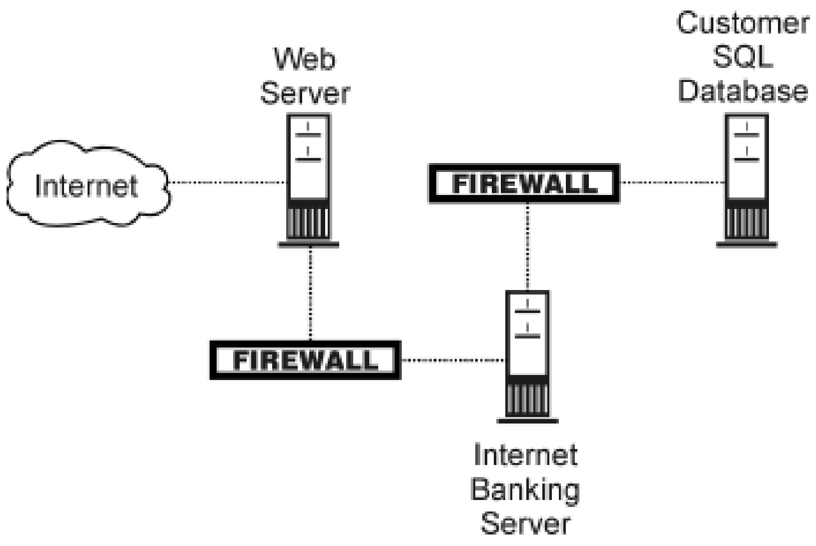
3. Get balance information

4. Withdrawal of money

5. Transfer Money

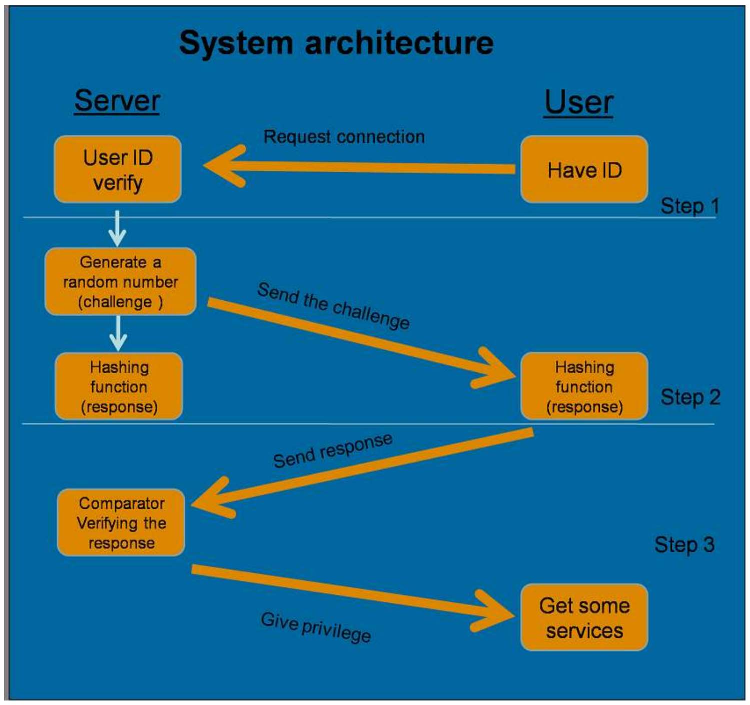
6. Customer info.

3.1 Login:



Customer logins by entering user-id& a password.

3.2 Validation:



When a customer enters the ATM card, its validity must be ensured. Then customer is allowed to enter the valid PIN. The validation can be for following conditions

Validation for lost or stolen card

When card is already reported as lost or stolen then the message “Lost/Stolen card!!!”.

Validation for card’s expiry date

If the card inserted by the customer has crossed the expiry date then the system will prompt

“Expired Card”.

Validation for PIN :

After validating the card, the validity of PIN must be ensured. If he/she fails to enter valid code for three times then the card will not be returned to him. That means the account can be locked. The counter for number of logins must be maintained

Get balance information:

This system must be networked to the bank’s computer. The updated

database of every customer is maintained with bank. Hence the balance information of every account is available in the database and can be displayed to the customer.

3.3 Payment of Money:

A customer is allowed to enter the amount which he/she wishes to withdraw. If the entered amount is less than the available balance and if after withdraw if the minimum required balance is maintained then allow the transaction.

3.4 Transfer of Money:

The customer can deposit or transfer the desired amount of money.

3.5 Transaction Report:

The bank statement showing credit and debit information of corresponding account must be printed by the machine.

3.6 Technical Issues

This product will work on client-server architecture. It will require an internet server and which will be able to run PHP applications. The product should support some commonly used browsers such as Internet Explorer, Mozilla Firefox.

4. Interface Requirements

4.1 GUI

This is interface must be highly intuitive or interactive because there will not be an assistance for the user who is operating the System. At most of the places help desk should be provided for users convenience. The screens appearing should be designed in such a manner that it can draw User attaraction towards the new plans for the customers.

Also the pin and password confidentiality should be maintained,

This can be done by using asterisks at the password panel.

Proper security messages should be displayed at most of the places.

4.2 Hardware Interface

Various interfaces for the product could be

1. Touch screen/Monitor
2. Keypad
3. Continuous battery backup
4. Printer which can produce the hard copy.
5. Interface that connects the device to bank's computer.
6. An interface that can count currency notes.

4.3 Software Interface

1. Any windows and mac operating system.

2. The PHP must be installed. For the database handling MYSQL must be installed. These products are open source products.

3. The final application must be packaged in a set up program, so that the products can be easily installed on machines. This application must be networked to corresponding banks.

5. Performance Requirements

The system should be compatible enough to hold the general traffic .

It should not get hang or show some other problems arising out due to large no of concurrent users . The system should be fast enough to meet the customer The high and low temperature should not affect the performance of the device. An uninterrupted transaction must be performed.

6.Constraints

* The information of all the users must be stored in a database that is accessible by the Online Banking System.

* The Online Banking System is connected to the computer and is running all 24hours a day.

* The users access the Online Banking System from any computer that has Internet browsing capabilities and an Internet connection.

*The users must have their correct usernames and passwords to enter into the Online Banking System.

Design Constraints:

*** Software Language Used**

The languages that shall be used for coding Online Banking System are c , c++ , java , and HTML. For working on the coding phase of the Online job portal System Web Sphere Application Server/WebSphere Application Server CE Server needs to be installed.

***Database design**

In our HDFC BANK database design, we give names to data flows, processes and data stores. Although the names are descriptive of data, they do not give details .So following DFD, our interest is to build some details of the contents of data flows, processes and data store. A data dictionary is a structured repository of data about data .It is a set of rigorous definitions of all DFD data elements and data structures .

7. Performance

7.1 Security

The banking system must be fully accessible to only authentic user.

It should require pin for entry to a new environment.

7.2 Reliability

The application should be highly reliable and it should generate all the updated information in correct order.

7.3 Availability

Any information about the account should be quickly available from any computer to the authorized user. The previously visited customer's data must not be cleared.

7.4 Maintainability

The application should be maintainable in such a manner that if any new requirement occurs then it should be easily incorporated in an individual module.

7.5 Portability

The application should be portable on any windows based system. It should not be machine specific.

Definitions:

- **Account**

A single account in a bank against which transactions can be applied. Accounts may be of various types with at least checking and savings. A customer can hold more than one account.

- **ATM**

A station that allows customers to enter their own transactions using cash cards as identification. The ATM interacts with the customer to gather transaction information, sends the transaction information to the central computer for validation and processing, and dispenses cash to the customer. We assume that an ATM need not operate independently of the network.

- **Bank**

A financial institution that holds accounts for customers and that issues cash cards authorizing access to accounts over the ATM network.

- **Bank computer**

The computer owned by a bank that interfaces with the ATM network and the bank's own cashier stations. A bank may actually have its own internal network of computers to process accounts, but we are only concerned with the one that interacts with the network.

- **Cash Card**

A card assigned to a bank customer that authorizes access to accounts using an ATM Machine. Each card contains a bank code and a card number, coded in accordance with national standards on credit cards and cash cards. The bank code uniquely identifies the bank within the consortium. The card number determines the accounts that the card can access. A card does not necessarily access all of a customer's accounts. Each cash card is owned by a single customer, but multiple copies of it may exist, so the possibility of simultaneous use of the same card from different machines must be considered.

- **Customer**

The holder of one or more accounts in a bank. A customer can consist of one or more persons or corporations, the correspondence is not relevant to this problem. The same person holding an account at a different bank is considered a different customer.

- **Transaction**

A single integral request for operations on the accounts of a single customer. We only specified that ATMs must dispense cash, but we should not preclude the possibility of printing checks or accepting cash or checks. We may also want to provide the flexibility to operate on accounts of different customers, although it is not required yet. The different operations must balance properly.

8 References:

www.w3schools.com

www.roseindia.net

www.dbforums.com

www.ibm.com

http://tomcat.apache.org/

Because of the speed, flexibility, and efficiency that it offers, the Internet has become the means for conducting growing numbers of transactions between suppliers and large international corporations. In this way, the Internet has opened new markets to the world and has accelerated the diffusion of knowledge. The meaning of Internet markets or online business has been widely used in these days. The success of the business depends on its flexibility, availability and security. Since that the web-based systems should have a special way to design the system and implement it. Nowadays, the Internet Banking System widely used and the banks looking to provide the best quality system with highly available, fast response, secure and safe to use. The Unified Modelling Language (UML) is the uniquely language which is used to analyse and design any system. In this paper, the UML diagrams has been proposed to illustrate the design phase for any banking system. The authors, presented two types of architecture which is used for the Internet Banking System.

1. INTRODUCTION In the recent years there has been explosion of Internetbased electronic banking applications (Liao & Cheung, 2003). Beckett, Hewer & Howcroft (2000) states that the emergence of new forms of technology has created highly competitive market conditions for bank providers. However, the changed market conditions demand for banks to better understanding of consumers' needs [1]. Liao et al. (2003) stress that the success in Internet banking will be achieved with tailored financial products and services that fulfill customer' wants, preferences and quality expectations. Mattila (2001) concedes that customer satisfaction

is a key to success in Internet banking and banks will use different media to customize products and services to fit customers' specific needs in the future. Liao et al. (2003) suggest that consumer perceptions of transaction security, transaction accuracy, user friendliness, and network speed are the critical factors for success in Internet banking. From this perspective, Internet banking includes many challenges for human computer interaction (HCI) [2]. Hiltunen et al (2004) have remarked that there are at least two major HCI challenges in Internet banking. The first challenge is related to the problem how to increase the number of services of Internet banking and simultaneously guarantee the quality of service for individual customers [3]. The second challenge is related to the problem how to understand customer's needs, translate them into targeted content and present them in a personalized way in usable user interface. Hiltunen et al. (2004) imply that Internet banking research will concentrate more on HCI factors in the future [4]. Recently, Lindgaard & Dudek (2003) emphasize that now is an ideal time for HCI researchers to analyse user satisfaction, because there is growing interest in how to attract and increase the number of online customers in ebusiness and e-commerce. Lindgaard et al. (2003) stress that HCI researchers should reveal a structure of user satisfaction, determine how to evaluate it and conclude how it is related to the overall user experience of online customers. The concept of electronic banking has been defined in many ways (e.g. Daniel, 1999). According to Karjaluoto (2002) electronic banking is a construct that consists of several distribution channels. Daniel (1999) defines electronic banking as the delivery of banks' information and services by banks to customers via different delivery platforms that can be used with different terminal devices such as a personal computer and a mobile phone with browser or desktop software, telephone or digital television [5]. The system models are abstract view of a system that ignores some system details. Complementary system models can be developed other information about the system. And they are graphical representations that describe business processes, the problem to be solved and our system is to be developed. This reading may be difficult to interpret. It is presented to cover the entire concept of process modeling using the specific language of systems design. In the homework for this module and in a separate module reading (the Demonstration Project), you will see these concepts applied to real situations which will help you integrate the concepts [6]. System models play an important role in systems development. Because system analysts are dealing with unstructured problems (and end-users, too, deal with them), there needs to be a systematic, logic-based approach to converting a real-world problem, no matter how vague, into a representation (or model) that captures the main points and relationships so that the problem can be analyzed. A model is a representation of the designer's interpretation of reality. Models can be building for existing systems as a way to understand better those systems, or for proposed systems as a way to document the organizational and information requirements or technical designs [7].

2. CONTEXTS DIAGRAM A System Context Diagram is the highest level view of a system, similar to Block Diagram, showing a (normally software-based) system as a whole and its inputs and outputs from/to external factors. The Context Diagrams show the interactions between a system and other actors with which the system is designed to face. They are also typically drawn using labeled boxes to represent each of the external entities and another labeled box to represent the system being developed. The relationship is drawn as a line between the entities and the system being developed [7]. Context Diagram is a data flow diagram showing data flows between a generalized application within the domain and the other entities and abstractions with which it communicates. One thing that differentiates the use of data flow diagrams in domain analysis from other typical uses is that the variability of the data flows across the domain boundary must be accounted for with either a set of diagrams or text describing the differences [8]. Before we construct the actual process model, we need to establish initial project scope. A project is scope defines what aspect of the business a system or application is supposed to support. It also defines how the system or application being modeled must interact with other systems and the business as a whole. A project's scope is documented with a context diagram

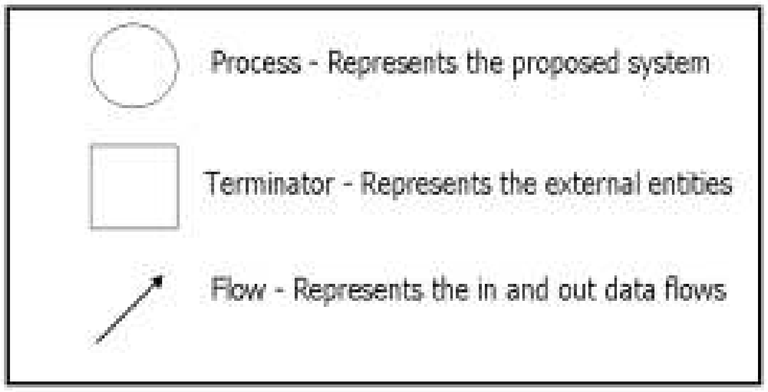

Process - Represents the proposed system
Terminator - Represents the external entities
Flow - Represents the in and out data flows

Fig 1. Notation for context Diagram

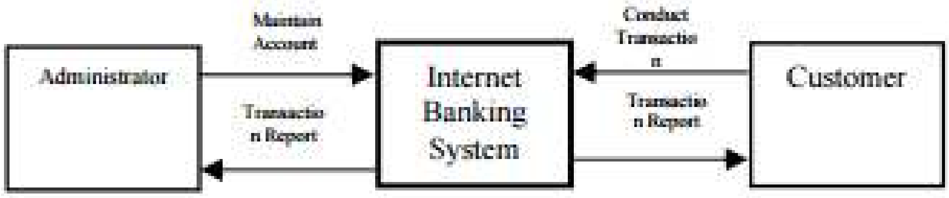


Fig 2. Context Diagram of internet banking system

[8].

3. SEQUENCE DIAGRAM UML sequence diagrams model the flow of logic within your system in a visual manner, enabling you both to document and validate your logic, and are commonly used for both analysis and design purposes. Sequence diagrams are the most popular UML artifact for dynamic modeling, which focuses on identifying the behavior within your system. Sequence diagrams, along with class diagrams and physical data models are in experts' opinion the most important design-level models for modern business application development. The Message Sequence Chart technique has been incorporated into the Unified Modeling Language (UML) diagram under the name of Sequence Diagram. A sequence diagram shows, as parallel vertical lines, different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner. The dotted lines extending downwards indicate the timeline, time flows from top to bottom. The arrows represent messages (stimuli) from an actor or object to other objects [9]. The UML 2.0 Sequence Diagram supports similar notation to the UML 1.x Sequence Diagram with added support for modeling variations to the standard flow of events. If the lifeline is that of an object, it is underlined (if not it is a role). Note that leaving the instance name blank can represent anonymous and unnamed instances. In order to display interaction, messages are used. These are horizontal arrows with the message name written above them. Solid arrows with full heads are synchronous calls, solid arrows with stick heads are asynchronous calls and dashed arrows with stick heads are return messages. This definition is true as of UML 2, considerably different from UML 1.x. Activation boxes, or method-call boxes, are opaque rectangles drawn on top of lifelines to represent that processes are being performed in response to the message. Objects calling methods on themselves use messages and add new activation boxes on top of any others to indicate a further level of processing. When an object is destroyed (removed from memory), an X is drawn on top of the lifeline, and the dashed line ceases to be drawn below it. It should be the result of a message, either from the object itself, or another. A message sent from outside the diagram can be represented by a message originating from a filled-in circle. A UML diagram may perform a series of steps, called a super step, in response to only one external stimulus [10]. Sequence diagrams are typically used to model:

• Usage scenarios. A usage scenario is a description of a potential way your system is used. The logic of a usage scenario may be part of a use case, perhaps an alternate course. It may also be one entire pass through a use case, such as the logic described by the basic course of action or a portion of the basic course of action, plus one or more alternate scenarios. The logic of a usage scenario may also be a pass through the logic contained in several use cases. For example, a student enrolls in the university, and then immediately enrolls in three seminars [10]. • The logic of methods. Sequence diagrams can be used to explore the logic of a complex operation, function, or procedure. One way to think of sequence

• diagrams, particularly highly detailed diagrams, is as visual object code. • The logic of services. A service is effectively a highlevel method, often one that can be invoked by a wide variety of clients. This includes web-services as well as business transactions implemented by a variety of technologies such as CICS/COBOL or CORBAcompliant object request brokers (ORBs).

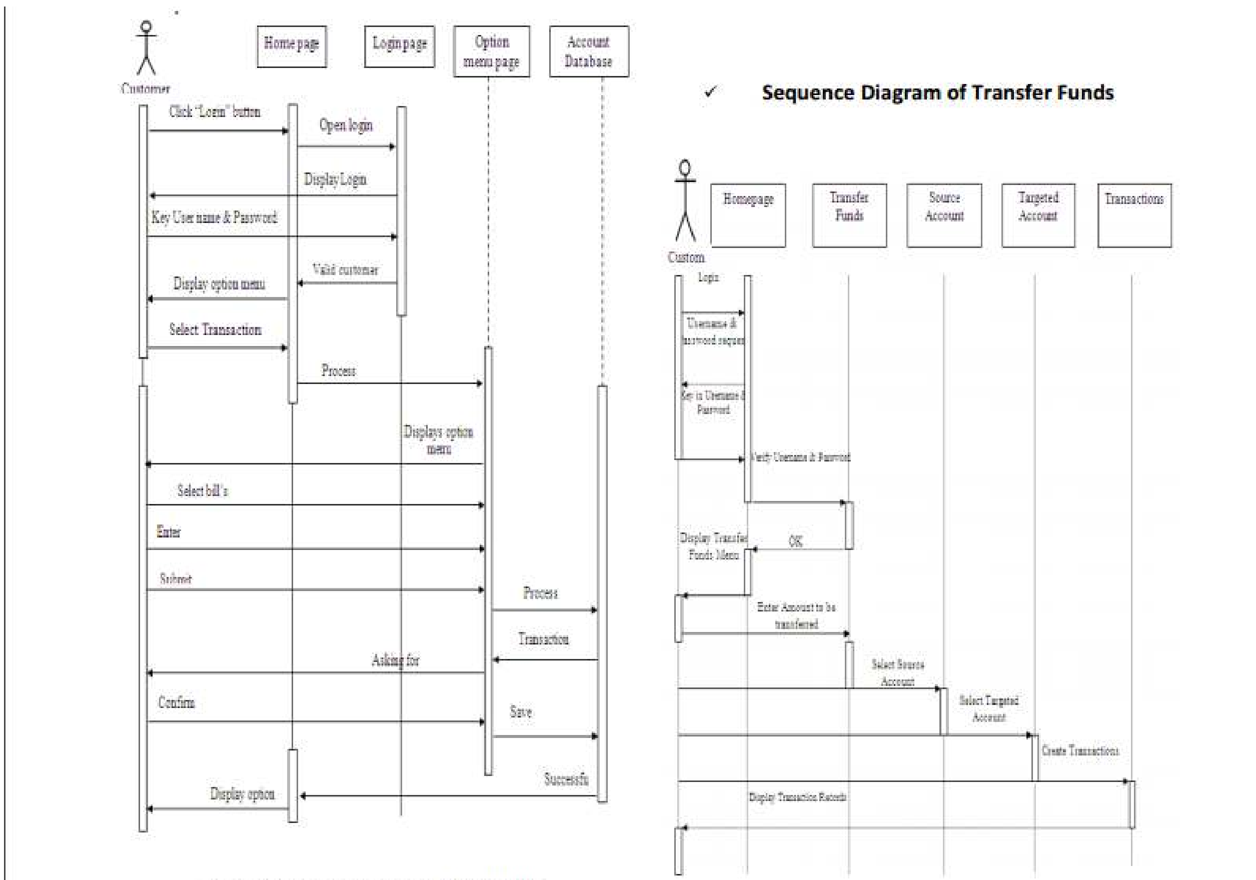


Fig 3. Sequence Diagram of "Pay Bills"

4. CLASS DIAGRAM Static models of a system describe the structural relationships that hold between the Pieces of data manipulated by the system. They describe how data is parcelled out into Objects, how those objects are categorized, and what relationships can hold between them. They do not describe the behavior of the system, nor how the data in a system evolves over time. These aspects are described by various types of dynamic model. The most important kinds of static model are object diagrams and class diagrams. An object diagram provides a 'snapshot' of a system, showing the objects that actually exist at a given moment and the links between them. Many different object diagrams can be drawn for a system, each representing the state of the system at a given instant. An object diagram shows the data that is held by a system at a given moment. This data may be represented as individual objects, as attribute values stored inside these objects, or as link between objects [11].

✓ **Sequence Diagram of Cheque Services**

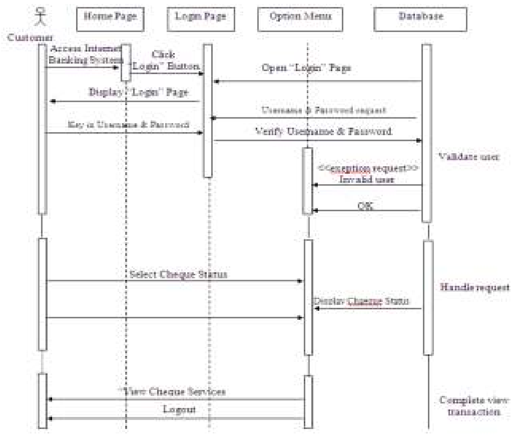


Fig 5. Sequence Diagram of "Cheque Services"

✓ **Sequence Diagram of View Account Transaction**

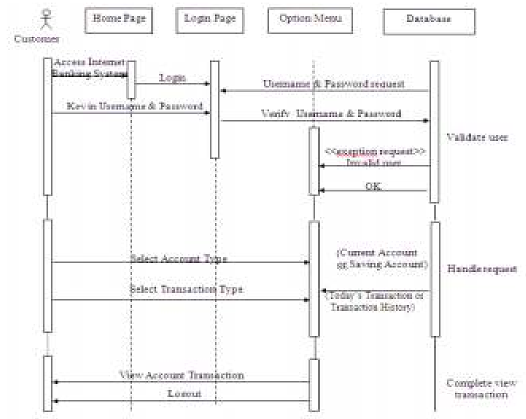
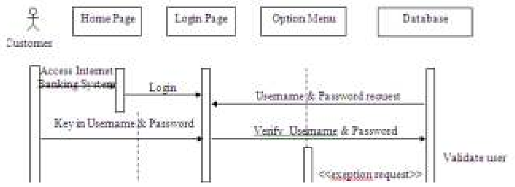

Customer
Home Page
Login Page
Option Menu
Database
Access Internet Banking System
Login
Username & Password request
Key in Username & Password
Verify Username & Password
Validate user
<<exeption request>> Invalid user
OK
Select Account Type
(Current Account or Saving Account)
Handle request
Select Transaction Type
(Today's Transaction or Transaction History)
View Account Transaction
Logout
Complete view transaction

Fig 6. Sequence Diagram of "View Account Transaction" Use case

✓ **Sequence Diagram of Utilities**


Customer
Home Page
Login Page
Option Menu
Database
Access Internet Banking System
Login
Username & Password request
Key in Username & Password
Verify Username & Password
Validate user
<<exeption request>>

6. ARCHITECTURAL DESIGN The architecture embodies the major static and dynamic aspects of a system. It is a view of the whole system highlighting the important characteristics and ignoring unnecessary details. In the context of our approach, architecture is primarily specified in terms of views of five models; the Use-Case model, Analysis Model, Design Model, Deployment model and Implementation model. These views show the "architecturally significant" elements of those models. The models have the following specific characteristics in our approach: The Use-Case model shows the thematic use cases related to functionality associated with distribution The Analysis model illustrates how boundary, control and entity classes are associated with the thematic use cases identified in the Analysis. Remote Communication Control classes shown in this model are specializations of Control classes and represent the abstraction of components that deal with remote communication and distribution using CORBA (Common Object Request Broker Architecture). The Design model shows the design classes that trace the specialized Remote Communication Control classes in analysis. Special attention is given to the interfaces provided by these design classes. We show how some of these are represented by IDL interfaces. The Implementation model describes how elements in the design model are implemented in

term of components. Finally, the Deployment model explains how CORBA-based components are assigned to nodes [12]

Two proposed Architectures for internet Banking System 1. In a broader term, Architecture should not only focus on applications but it should also focus on Information Technology Service Management as a whole. It includes Technology, Processes, People and Information. Large data intensive system typically consists of a set of cooperating autonomous subsystems that brings the concept of multiple architectures within a system. In the following sections, we for large systems such as banking system to have an organizing concept that works for all modules within the system. In this way we can easily figure out the qualities that the proposed system should have in the required components. This will help to make the architecture more clear and understandable. 2. Within the architecture layer, we use different views to enhance the understandability of the architecture so that we can focus on particular concerns separately. We conceptual, logical and execution views, as described below.

6.1 1ST ARCHITECTURE Architecture Design involves in an early stage of the system design process represents the link between and processes. It involves identifying major system components and their communications The primary goals of architecture documentation are to Record the architects decisions in documentation. To meet this goal, the documentation must be complete and clear. Communicate the architecture. To meet this goal, consider what each stakeholder of the system needs to know, and what are the best procedures to convey what they need to know. The comprehensive architecture specification document can address this goal.

6.2 2ND ARCHITECTURE The Conceptual Architecture identifies the high-level components of the system and the relationships among them. The purpose of this architecture is to direct attention at an appropriate decomposition of the system without discussing details. This view provides useful methods for communicating the architecture to nontechnical audiences, such as management, marketing, and users. It consists of the Architecture Diagram (without interface detail) and an informal component specification for each component. Modular Decomposition Modular decomposition is the process that the identified sub-systems are decomposed into sub modules.

7. JUSTIFICATIONS OF THE PROPOSED ARCHITECTURES The software architecture of a program or computing system is the structure of the system, which comprises software elements, the visible properties of those elements, and the relationships among them. Software architecture can be defined in many ways. In software architecture fundamental organization of a system embodies their relationships to each other and the environment in its components along with the principles governing design and evolution. An architecture can also be defined as a set of significant decisions about the organization of a software system, the selection of the structural elements, and their interfaces by which the system is composed, together with their behavior as specified in the collaborations among those elements. The composition of these structural and behavioral elements into progressively larger subsystems and the architectural style guides the organization, the mentioned elements, their interfaces, collaborations, and composition. Though Architecture drivers are not part of the architecture as such, the drivers that shape the architecture are important. They include: Architecture Vision: expressing the

desired state that the architecture will bring about. Architectural Requirements: capturing stakeholder goals and architecturally significant behavioral (functional) requirements as well as system qualities (non-functional requirements) and constraints. Assumptions, Forces and Trends: documenting assertions about the current business, market and technical environment now and over the architecture planning horizon[13]. 7.1 AVAILABILITY Quality attributes of large software system

7.2 SECURITY Security is possibly the most significant barrier to acceptance of IT services and digital services as a utility becoming absolutely crucial in a more dynamic often justify their creation by claiming that it supports and promotes certain qualities, often called non- functional qualities. These qualities including portability, reusability, performance, modifiability, and scalability are supposed to be automatically conferred on any system that is realized using the architecture. The achievement of non-functional qualities is attributable to many factors (such as coding styles, documentation, testing, etc.). However, larger the system, the more the achievement of non-functional qualities rests in a system's software architecture. In large systems a Meta Architecture approach will improve both the functional and non functional quality attributes [14] 8. CONCLUSION Bank-Focused system model, though less risky, does not offer much when it comes to extending financial service outreach to the poor and unbanked. Both Bank-Led and Nonbank-Led system models offer a greater potential to achieve this objective. These system models, however, vary in their potential as well as risks. The decision as to which model must be adopted should be made after carefully weighing the risk-return tradeoff. A careful approach may be adopted to start with the less risky bank-led model and gradually adding more options as the players and stakeholders become more experienced. Once a model of branchless banking is decided upon, work towards creating an enabling regulatory environment for implementation of that model should start. Many components of such an environment are already in place if bank-led system model is adopted. However, Clear guidelines regarding various aspects of allowable activities should be issued to avoid uncertainties. Further, a forceful eradication of any unlawful and unauthorized services and offerings (generally provided by unlicensed players) - which may sprout up - is a must to promote and safeguard the interest of genuine players and the overall system. Banking systems usually contains legacy systems along with very large database systems. For internet banking applications a large number of interfaces are incorporated to facilitate the customers especially in consumer banking applications. Handling of financial transactions requires taking care of various issues including authentication, consumer privacy, money laundering, liability for unauthorized transactions.