**C h a p t e r 1**

**A b s t r a c t**

The project is to develop an standalone Loan Application Processing System (LAPS). This document contains the work flow of the system. Most of the bank out-sources pre-loan process to loan agencies to reduce the burden and let the agencies pick up the information from customers and verify it before it is being forwarded to the actual bank for approval of loan. This is an interface which facilitates a customer to apply for a loan from on-line and to track the status from time-to-time along with aiding the loan approval agency to verify and accept/reject the customer file. This is a way, it not only helps the customers but also the loan agency to check the pending, assign it to a departments, complete the formalities and procedures between the departments and arrive at decisions to very fact in addition to providing a transparency system for everyone. The customer can directly apply for a loan by selecting a bank and loan type from the list available. The application is received by lad who will have three departments Pickup, Verification and Legal.

This system can be controlled by the administrator. First he will look at the application received and allot the application for a particular employee of pickup department. The employee will go and make a physical verification of the documents at the customers and receives the documents necessary for the loan. Then he logs into this system and forwards the application to the verification department which will verify the whereabouts of the person, his organization, his salary particulars etc. and then forwards the application with a status verified. Then application reaches the legal department. The legal department people will verify the builder details and when satisfied sends their report to the administrator. The administrator or final approving authority views both types of reports, Via, the reports from verification department and the report legal department. This will help him to take a decision regarding whether to forward it to the bank or not. The same is communicated to the customer. The customer can at any time view the status of his application and can send any messages to the administrator and can get clarifications from him. Thus the software helps to simplify the loan system along with making the work easy.

There are 3 types of users who can get immense benefits from system:

• The customer - seeking the loan and information related to banks and loans

• The administrator of loan agency who will take track the decision of bank to approve or disapprove and also controls the overall system functionality

• The Loan approval department(lad) users who picks up the details and documents from customers and make a physical verification of the details submitted by the customer and the legality of the documents of the builder and construction.

\*\*\*\*\*\*Technologies Used: Servlets, JDBC and Java Script. \*\*\*\*\*\*\*\*\*\*\*

**Chapter 2**

**Existing System**: Here the existing system is a manual one using which the banking agent can’t maintain the effectively by sharing across different branches with proper security and can’t track details easily. It doesn’t provide proper co-ordination between different departments of the company. It doesn’t allow the customer to check the status of his file in proper way which leads customer dis-satisfaction.

¬ Doesn’t provide faster and effective system

¬ Doesn’t provide good co-ordination between departments

¬ Doesn’t provide effective forwarding system to move the file from one level to another

¬ Doesn’t user-friendly interface

¬ Difficulty in generating different reports as per the business requirement

¬ Doesn’t facilitate the services from online

**Proposed System**:

The online automated system with web-based architecture can support issues like.

¬ This system maintains the information related different departments and stored at a central DB, which leads easy accessibility and consistency

¬ The decision process is faster and more consistent

¬ Provides good communication between two departments

¬ Provides a facility to generate the reports very easily

**Chapter 3**

**FEASIBILITY REPORT**:

Feasibility study is the high level capsule version of the entire requirement analysis process. The objective of feasibility study is to determine whether the proposed system can be developed with available resources. There are three steps to be followed for determining the feasibility study of proposed system.

¬ Technical Feasibility

¬ Operational Feasibility

¬ Economical Feasibility

**Technical Feasibility**:

It is concerned with hardware and software feasibility. In this study, one has to test whether the proposed system can be developed using existing technology or not. If new technology is required ,what is the likely hood that it can be developed ?The organization for which the system to be developed is not provided online services .Hence there is a requirement of new hardware and software technology for the deployment of proposed system. As per client requirements the system to be developed should have speed response because of fast change info, programming productivity, reliability, security, scalability, integration and availability.

\*\*\*\*\*\*\*\*\*To meet these requirements I as a developer found jsp1.1 as a right choice because of its features platform independent, modularity and reusability. \*\*\*\*\*

**Operational Feasibility**:

Operational feasibility determines whether the proposed system satisfied the user objectives and can be fitted in to current system operation. The proposed system can be justified as operationally feasible basing on the following.

¬ The methods of processing and presentation are completely acceptable by the clients because they meet all the user and client requirements.

¬ The clients have been involved during the preparation of requirement analysis and design process.

¬ The system will certainly satisfy the user objectives and it will also enhance their capability.

¬ The system can be best fitted into current operation and requires little training to both administrator and dealer.

With the help of this system customer to place order requires simple data entry through forms provided. The proposed system is completely user friendly.

**Economical Feasibility**:

This includes an evaluation of all incremental costs and benefits expected if proposed system is implemented. cost-benefit analysis which is to be performed during economical feasibility delineates costs for project development and weighs them against benefits of system. In this the proposed system replaces the manual process of receiving orders which benefits the organization to get more orders and good response. So developing this system is economically feasible to organization.

**Application design**

There are three categories of people who would access the system viz. customers (loan applicants), members of **l**oan **a**pproval **d**epartment (lad) and administrators. Each one of them would have some exclusive privileges (for e.g. Customers can apply for a program by filling up the online form without any login process, members of **l**oan **a**pproval **d**epartment (lad) alone will be able to view loan applications, approve or reject the application for a specific loan program. Only the administrator has the right to keep track of the home finance provider’s loan program details.)

1. Customers should be able to
   * View all loan programs offered by the home finance provider
   * Apply online for a loan program offered by the home finance provider, by filling up the application form that auto generates the application ID
   * View the application status of application, based on the application ID
2. The member of **l**oan **a**pproval **d**epartment (lad) should be able to :
   * Login into the system using his/her credentials.
   * View all loan programs offered by the home finance provider
   * View loan applications for a specific loan program.
   * Accept/Reject an application on the basis of the details of the loan applicant. If loan application is acceptable, loan application status is changed to ‘Accepted’, the scheduled date for an interview (of the loan applicant for verification of applicant’s documents, before approving the applicant’s loan under the mentioned loan program) is filled into the application form.
   * After the interview, the status of the loan application is changed to Approved/Rejected
3. The administrators should be able to
   * login to the system using his/her credentials
   * Update and manage (add or delete) information of the loan programs offer by the home finance provider
   * Generate various reports like:
     + View List of loan applications approved/accepted (waiting for interview)/rejected for a loan program.
     + View all loan programs offered by the home finance provider

Transition of Status: Applied ->Accepted/Rejected->Approved/Rejected

\*\*\*\*\*\*\*\*\*\*\*\*UML DIAGRAMS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Number of Modules**: The system after careful analysis has been identified to present itself with the following modules

1. Administrator Module: This module is responsible for coordinating the other modules. It allows the administrator create, update/delete and view the banks information and it allows admin to create create/update/delete and view different departments and it can create logins for different employees in each and every department and it can manage loan interest rates of different banks etc. It facilitates to view the new applicant details and assign it to different employees in pickup department initially. It allows the administrator to view customer application finally from verification department and builder details from legal department and then store bank final opinion in this application. It also allows to generate different reports for business analysis. It provides messaging facility for the administrator for communication.

2. lad Module: This module allows the pickup department to view their applications which are assigned to them, collect the documents according to the checklist and forward it to verification department. This module allows the employees of verification department employees to view the forwarded application from pickup department and check the details as per the documents and forward it to next level.This module allows the employee of legal departments to check verify legal documents of the builder, verify the check list and then generate the APF no for the builder.

3. Customer Module: This module allows the customer to view the interest rates of the banks which we are dealing, apply for a loan, check the status of the loan at any point of time and communicate with the administrator if necessary. It allows messaging facility for communication.

**Requirements at the Server-side**

Hard Ware requirements:

* Intel Pentium 90 or higher (P166 recommended)
* Microsoft Windows 95, 98, or NT 4.0, 2k, XP, Windows 7
* Memory: 32MB of RAM (64MB or more recommended)

**Software specification**

* Internet Explorer 6.0 or higher
* Oracle 9i client and access to oracle 9i server
* JDK 8
* Eclipse Luna
* JUnit 4.0,log4j

## \*\*\*\*\*\*\*\*\*\*\*\*\*\*Technology used:

* + - *Front End & Web Components:–* 
      1. HTML/JavaScript
      2. Servlets
      3. JSP
    - *Business Logic Components and Services :-* 
      1. Java Beans
    - *Application Servers:-* 
      1. WildFly
    - *Databases:-*
      1. Oracle 9i\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Chapter 4**

Analysis Report SRS Document Intended Audience And Reading Suggestions

To make the application building process

¬ To minimize the debugging problems

¬ To generate bug free code from high level of expressions of requirements

¬ To make languages easy to use and understand

**Over view**: \*\*\*\*\*\*\*\*about software\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

With the varied topic in existence in the fields of computers, Client Server is one, which has generated more heat than light, and also more hype than reality. This technology has acquired a certain critical mass attention with its dedication conferences and magazines. Major computer vendors such as IBM and DEC, have declared that Client Servers is their main future market. A survey of DBMS magazine revealed that 76% of its readers were actively looking at the client server solution. The growth in the client server development tools from $200 million in 1992 to more than $1.2 billion in 1996. Client server implementations are complex but the underlying concept is simple and powerful. A client is an application running with local resources but able to request the database and relate the services from separate remote server. The software mediating this client server interaction is often referred to as MIDDLEWARE. The typical client either a PC or a Work Station connected through a network to a more powerful PC, Workstation, Midrange or Main Frames server usually capable of handling request from more than one client. However, with some configuration server may also act as client. A server may need to access other server in order to process the original client request. The key client server idea is that client as user is essentially insulated from the physical location and formats of the data needs for their application. With the proper middleware, a client input from or report can transparently access and manipulate both local database on the client machine and remote databases on one or more servers. An added bonus is the client server opens the door to multi-vendor database access indulging heterogeneous table joins. What is a Client Server Two prominent systems in existence are client server and file server systems. It is essential to distinguish between client servers and file server systems. Both provide shared network access to data but the comparison dens there! The file server simply provides a remote disk drive that can be accessed by LAN applications on a file by file basis. The client server offers full relational database services such as SQL-Access, Record modifying, Insert, Delete with full relational integrity backup/ restore performance for high volume of transactions, etc. the client server middleware provides a flexible interface between client and server, who does what, when and to whom. Why Client Server Client server has evolved to solve a problem that has been around since the earliest days of **computing**, how best to distribute your computing, data generation and data storage resources in order to obtain efficient, cost effective departmental and enterprise wide data processing. During mainframe era choices were quite limited. A central machine housed both the CPU and DATA (cards, tapes, drums and later disks). Access to these resources was initially confined to batched runs that produced departmental reports at the appropriate intervals. A strong central information service department ruled the corporation. The role of the rest of the corporation limited to requesting new or more frequent reports and to provide hand written forms from which the central data banks were created and updated. The earliest client server solutions therefore could best be characterized as “SLAVE-MASTER”. Time-sharing changed the picture. Remote terminal could view and even change the central data, subject to access permissions. And, as the central data banks evolved in to sophisticated relational database with nonprogrammer query languages, online users could formulate adhoc queries and produce local reports without adding to the MIS applications software backlog. However remote access was through dumb terminals, and the client server remained subordinate to the Slave\Master.

Front end or User Interface Design The entire user interface is planned to be developed in browser specific environment with a touch of Intranet-Based Architecture for achieving the Distributed Concept. The browser specific components are designed by using the HTML standards, and the dynamism of the designed by concentrating on the constructs of the Java Server Pages. Communication or Database Connectivity Tier. The Communication architecture is designed by concentrating on the Standards of Servlets and Enterprise Java Beans. The database connectivity is established by using the Java Data Base Connectivity. The standards of three-tier architecture are given major concentration to keep the standards of higher cohesion and limited coupling for effectiveness of the operations. Features o f The Language Used In my project, I have chosen Java language for developing the code. About Java Initially the language was called as “oak” but it was renamed as “Java” in 1995.

The primary motivation of this language was the need for a platform independent (i.e., architecture neutral) language that could be used to create software to be embedded in various consumer electronic devices.

¬ Java is a programmer’s language.

¬ Java is cohesive and consistent.

Compiling and interpreting Java Source Code. During run-time the Java interpreter tricks the byte code file into thinking that it is running on a Java Virtual Machine. In reality this could be a Intel Pentium Windows 95 or SunSARC station running Solaris or Apple Macintosh running system and all could receive code from any computer through Internet and run the Applets. Simple Java was designed to be easy for the Professional programmer to learn and to use effectively. If you are an experienced C++ programmer, learning Java will be even easier. Because Java inherits the C/C++ syntax and many of the object oriented features of C++. Most of the confusing concepts from C++ are either left out of Java or implemented in a cleaner, more approachable manner. In Java there are a small number of clearly defined ways to accomplish a given task. Object-Oriented Java was not designed to be source-code compatible with any other language. This allowed the Java team the freedom to design with a blank slate. One outcome of this was a clean usable, pragmatic approach to objects. The object Source Code ……….. ……….. ……….. ………… PC Compiler Macintosh Compiler SPARC Compiler Java Byte code (Platform independent) Java Interpreter (PC) Java Interpreter (Macintosh) Java Interpreter (Sparc) model in Java is simple and easy to extend, while simple types, such as integers, are kept as high-performance non-objects. Robust The multi-platform environment of the Web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. The ability to create robust programs was given a high priority in the design of Java. Java is strictly typed language; it checks your code at compile time and run time. Java virtually eliminates the problems of memory management and de-allocation, which is completely automatic. In a well-written Java program, all run time errors can –and should –be managed by your program. Servlets, JSP, JDBC & HTML

So why not just use ODBC from Java? The answer is that you can use ODBC from Java, but this is best done with the help of JDBC in the form of the JDBC-ODBC Bridge, which we will cover shortly. The question now becomes

**Question : "Why do you need JDBC?"** \*\*\*\*\*\*\*\*hibernet\*\*\*\*\*to include\*\*\*\*\*\*\*\*

1. ODBC is not appropriate for direct use from Java because it uses a C interface. Calls from Java to native C code have a number of drawbacks in the security, implementation, robustness, and automatic portability of applications.

2. A literal translation of the ODBC C API into a Java API would not be desirable. For example, Java has no pointers, and ODBC makes copious use of them, including the notoriously error-prone generic pointer "void \*". You can think of JDBC as ODBC translated into an object-oriented interface that is natural for Java programmers.

3. ODBC is hard to learn. It mixes simple and advanced features together, and it has complex options even for simple queries. JDBC, on the other hand, was designed to keep simple things simple while allowing more advanced capabilities where required.

4. A Java API like JDBC is needed in order to enable a "pure Java" solution. When ODBC is used, the ODBC driver manager and drivers must be manually installed on every client machine. When the JDBC driver is written completely in Java, however, JDBC code is automatically installable, portable, and secure on all Java platforms from network computers to mainframes. Two-tier and Three-tier Models

The JDBC API supports both two-tier and three-tier models for database access. In the two-tier model, a Java applet or application talks directly to the database. This requires a JDBC driver that can communicate with the particular database management system being accessed. A user's SQL statements are delivered to JAVA Application JDBC DBMS Client machine DBMS-proprietary protocol Database server the database, and the results of those statements are sent back to the user. The database may be located on another machine to which the user is connected via a network. This is referred to as a client/server configuration, with the user's machine as the client, and the machine housing the database as the server. The network can be an Intranet, which, for example, connects employees within a corporation, or it can be the Internet. In the three-tier model, commands are sent to a "middle tier" of services, which then send SQL statements to the database. The database processes the SQL statements and sends the results back to the middle tier, which then sends them to the user. MIS directors find the three-tier model very attractive because the middle tier makes it possible to maintain control over access and the kinds of updates that can be made to corporate data. Another advantage is that when there is a middle tier, the user can employ an easy-to-use higher-level API which is translated by the middle tier into the appropriate low-level calls. Finally, in many cases the three-tier architecture can provide performance advantages. Until now the middle tier has typically been written in languages such as C or C++, which offer fast performance. However, with the introduction of optimizing compilers that translate Java byte code into efficient machine specific code, it is becoming practical to implement the middle tier in Java. This is a big plus, making it possible to take advantage of Java's robustness, multithreading, and security features. JDBC is important to allow database access from a Java middle tier. Java applet or Html browser Application Server (Java) JDBC DBMS Client machine (GUI) HTTP, RMI, or CORBA calls Server machine (business Logic) DBMS-proprietary protocol Database server

**JDBC Driver Types**

The JDBC drivers that we are aware of at this time fit into one of four categories:

¬ JDBC

¬ ODBC Bridge plus ODBC driver

¬ Native-API partly-Java driver

¬ JDBC-Net pure Java driver

¬ Native-protocol pure Java driver JDBC

-ODBC Bridge

If possible, use a Pure Java JDBC driver instead of the Bridge and an ODBC driver.

This completely eliminates the client configuration required by ODBC. It also eliminates the potential that the Java VM could be corrupted by an error in the native code brought in by the Bridge (that is, the Bridge native library, the ODBC driver manager library, the ODBC driver library, and the database client library).

**What Is the JDBC- ODBC Bridge?**

The JDBC-ODBC Bridge is a JDBC driver, which implements JDBC operations by translating them into ODBC operations. To ODBC it appears as a normal application program. The Bridge implements JDBC for any database for which an ODBC driver is available. The Bridge is implemented as the sun.jdbc.odbc Java package and contains a native library used to access

**ODBC:** The Bridge is a joint development of Intersolv and JavaSoft. Java Server Pages (JSP) Java server Pages is a simple, yet powerful technology for creating and maintaining dynamic-content web pages. Based on the Java programming language, Java Server Pages offers proven portability, open standards, and mature re-usable component model .The Java Server Pages architecture enables the separation of content generation from content presentation. This separation not eases maintenance headaches; it also allows web team members to focus on their areas of expertise

**ORACLE**

**Tables**:

* + 1. LoanApplication: Application\_id,application\_date(date),Loan\_program(varchar(10)),AmountofLoan(number),AddressofProperty(varchar(30)),AnnualFamilyIncome(number),DocumentProofsAvailable(varchar(50)),GuaranteeCover (varchar(20)),MarketValueofGuaranteeCover(number), Status(varchar(10)), DateOfInterview(date)
    2. CustomerDetails:Application\_ID(FK),Applicant\_name(varchar(20)),date\_of\_birth(date),marital\_status(varchar(10)),phone\_number(number), mobile\_number(number),CountofDependents(number),email\_ id(varchar(20)),
    3. LoanProgramsOffered:ProgramName(varchar(5)),description (varchar(20)),type(varchar(00)),durationinyears(number), minloanamount(number),maxloanamount(number),rateofinterest(number), proofs\_required(varchar(20))
    4. ApprovedLoans:Application\_ID(FK),Customer\_name(varchar(20),amountofloangranted(number),monthlyinstallment(number),yearstimeperiod(number),downpayment(number),rateofinterest(number),totalamountpayable(number)
    5. Users : login\_id(varchar(5), password(varchar(10)), role(varchar(5))

**Chapter 5**

**C o d i n g**

**C h a p t e r 6**

**Program Design Language**

• The program design language is also called as structured English or pseudocode. PDL is a generic reference for a design language PDL looks like a modern language. The difference between PDL and real programming language lies in the narrative text embedded directly within PDL statements.

The characteristics required by a design language are:

¬ A fixed system of keywords that provide for all structured constructs date declaration and modularity characteristics

¬ A free syntax of natural language that describes processing features.

¬ Date declaration facilities that should include both simple and complex data structures.

¬ Subprogram definition and calling techniques that support various nodes of interface description

PDL can be extended to include keywords for multitasking and/or concurrent processing interrupt handling, interposes synchronization the application design for which PDL is to be used should dictate the final form for the design language.

\*\*\*\*\*\*\*\*\*\*PSUEDOCODE\*\*\*\*\*\*\*\*\*\*\*

**Chapter 7**

**Testing & Debugging Strategies**

Levels of Testing In order to uncover the errors present in different phases we have the concept of levels of testing. The basic levels of testing are as shown below… Client Needs Requirements Design Code System Testing The philosophy behind testing is to find errors. Test cases are devised with this in mind. A strategy employed for system testing is code testing.

**Code Testing**: This strategy examines the logic of the program. To follow this method we developed some test data that resulted in executing every instruction in the program and module i.e. every path is tested. Systems are not designed as entire nor are they tested as single systems. To ensure that the coding is perfect two types of testing is performed or for that matter is performed or that matter is performed or for that matter is performed on all systems.

\*Acceptance Testing

\*System Testing

\* Integration Testing

\*Unit Testing

**Unit testing**:

Focuses verification effort on the smallest unit of software i.e. the module. Using the detailed design and the process specifications testing is done to uncover errors within the boundary of the module. All modules must be successful in the unit test before the start of the integration testing begins. In this project each service can be thought of a module. There are so many modules like Login, HW Admin, Master Admin, Normal User, and P Manager. Giving different sets of inputs has tested each module. When developing the module as well as finishing the development so that each module works without any error. The inputs are validated when accepting from the user. In this application developer tests the programs up as system. Software units in a system are the modules and routines that are assembled and integrated to form a specific function. Unit testing is first done on modules, independent of one another to locate errors. This enables to detect errors. Through this error resulting from interaction between modules initially avoided.

\*\*\*\*\*\*\*\*\*\*Develop test classes for testing the following functionality

* + 1. Login
    2. Apply for Loan.
    3. Retrieve Loan Application Status

Test the application using JUnit. \*\*\*\*\*\*\*\*\*\*\*\*

**C h a p t e r 8**

**U s e r M a n u a l**

**Installation**

• The database as it is developed by Oracle11g can be installed only by creating the tables

• Copy the database driver insideprogram.

• Start the Server

• Open a browser and give the URL to access the application