Chapter-1

Abstract

ABSTRACT

The Domain “SunZip” lets you reduce the overall number of bits and bytes in a file so it can be transmitted faster over slower Internet connections, or take up less space on a disk. Domain Sun Zip is a System Based Software. The user need not depend on third party software’s like Winzip, Winrar etc.

The main algorithms are:

- Huffman algorithm

ALGORITHMS FOR HUFFMAN COMPRESSION / DECOMPRESSION

Huffman coding is an entropy encoding algorithm used for lossless data compression. The term refers to the use of a variable-length code table for encoding a source symbol (such as a character in a file) where the variable-length code table has been derived in a particular way based on the estimated probability of occurrence for each possible value of the source symbol. It was developed by David A. Huffman while he was a Ph.D. student at MIT, and published in the 1952 paper "A Method for the Construction of Minimum-Redundancy Codes".

Huffman coding uses a specific method for choosing the representation for each symbol, resulting in a prefix code (sometimes called "prefix-free codes") (that is, the bit string representing some particular symbol is never a prefix of the bit string representing any other symbol) that expresses the most common characters using shorter strings of bits than are used for less

common source symbols. Huffman was able to design the most efficient compression method *of* this type: no other mapping of individual source symbols to unique strings of bits will produce a smaller average output size when the actual symbol frequencies agree with those used to create the code. A method was later found to do this in linear time if input probabilities (also known as *weights*) are sorted.

For a set of symbols with a uniform probability distribution and a number of members which is a power of two, Huffman coding is equivalent to simple binary block encoding, e.g., ASCII coding. Huffman coding is such a widespread method for creating prefix codes that the term "Huffman code" is widely used as a synonym for "prefix code" even when such a code is not produced by Huffman's algorithm.

Although Huffman coding is optimal for a symbol-by-symbol coding (i.e. a stream of unrelated symbols) with a known input probability distribution, its optimality can sometimes accidentally be over-stated. For example, arithmetic coding and LZW coding often have better compression capability. Both these methods can combine an arbitrary number of symbols for more efficient coding, and generally adapt to the actual input statistics, the latter of which is useful when input probabilities are not precisely known or vary significantly within the stream. In general, improvements arise from input symbols being related (cat is more common than cta).

Chapter-2

Organization Profile

ORGANIZATION PROFILE

NIIT is a leading Global Talent Development Corporation, building skilled manpower pool for global industry requirements. The company which was set up in 1981, to help the nascent IT industry overcome its human resource challenges, has today grown to be amongst world's leading talent development companies offering learning solutions to Individuals, Enterprises and Institutions across 40 countries.

NIIT's training solutions in IT, Business Process Outsourcing, Banking, Finance and Insurance, Executive Management Education, and Communication and Professional Life Skills, touch five million learners every year. NIIT's expertise in learning content development, training delivery and education process management make it the most preferred training partner, worldwide.

Research-based Innovation, a key driver at NIIT, has enabled the organisation to develop programs and curricula that use cutting-edge instructional design methodologies and training delivery. NIIT's Individual Learning Solutions include industry-endorsed IT training programs like GNIIT, Integrated programs for Engineers (NIIT Edgeineers) and Infrastructure Management programmes (NIIT GlobalNet+).

For working professionals, NIIT Imperia, Centre for Advanced Learning, brings Executive Management Education Programs from premier B-schools in India, to their doorstep.

NIIT Institute of Finance Banking & Insurance (IFBI), formed by NIIT with equity participation from ICICI Bank, offers programs for individuals and corporates in Banking, Financial Services and Insurance.

NIIT Uniqua, Centre for Process Excellence, addresses the increasing demand for skilled workers in the business and technology services industry by providing training programs in relevant areas. This is a part of NIIT Institute of Process Excellence, a NIIT-Genpact venture.

NIIT's School Learning Solutions offers turnkey IT integration program for schools and has provided computer – based learning in over 12,000 government and private schools. NIIT eGuru, is a comprehensive learning solutions for schools. To address the vast population of underprivileged, school-aged children, NIIT launched the Hole-in-the-Wall education initiative. Its achievements in the area of Minimally Invasive Education earned NIIT the coveted Digital Opportunity Award, by the World Information Technology Services Alliance (WITSA) in 2008.

NIIT's Corporate Learning Solutions offers integrated learning solutions, including strategic consulting, learning design, content development, delivery, technology, assessment and learning management to Fortune 500 companies, Universities, Technology companies, Training

corporations and Publishing houses. Element K, delivers learning solutions for customers and partners through a tailored combination of catalog learning products, technology, and services. The offerings include: vLab®: hands-on labs, instructor-led courseware, comprehensive e-reference libraries, technical journals, and KnowledgeHub™ , a hosted learning management platform. NIIT, together with Element K, is now the first and the best choice for comprehensive learning solutions, worldwide.

Ushering in a new model in higher education is the not-for-profit NIIT University, established in 2009 with a vision of being the leading centre of innovation and learning in emerging areas of the Knowledge Society. Nestled in the foothills of Aravali, in Neemrana, Rajasthan, the picturesque 100 acres fully residential green campus has been developed as an institute of excellence based on the four core principles of providing industry linked, technology based, research driven, seamless education.

Chapter-3

Project Introduction

PROJECT INTRODUCTION

3.1. PURPOSE

The Domain “SunZip” lets you to reduce the overall number of bits and bytes in a file so it can be transmitted faster over slower Internet connections, or take up less space on a disk. Domain SunZip is a System Based Software. The user need not depend on third party software’s like winzip, winrar, Stuff etc. The main algorithm used for Compreision and Decompression is Huffman algorithm.

3.2. SCOPE

SunZip software is an open source product. It maintains a unique extension and security. Whenever user wants to open a compressed file made by SunZip, it requires software. It is available for all platforms like Windows, Linux, Macintosh etc. it provides a jar file for user, it act as an executable for all Operating System

3.3. OVERVIEW

The Domain “File Compression” lets you to reduce the overall number of bits and bytes in a file so it can be transmitted faster over slower Internet connections, or take up less space on a disk. Domain File compression is a System Based Software. The software will be done using Core Java. It can use in the System as a utility. The type of compression we will use here is called lossless compression. The user need not depend on third party software’s like winzip, winrar, Stuff etc. the software can be used to compress

files and they can be decompressed when the need arises. For implementing this Software we want to use Huffman algorithm

The Domain File Compression mainly include 5 modules

1. Compress the File Or Folder
2. De-Compress the File or folder
3. View files in the compressed file
4. Facility to set icon
5. Facility to set your own extension

1. COMPRESSING THE FILE OR FOLDER

This module helps us to compress a file or folder. The compressed file will have a extension that has been given at the development time. We can send the compressed file over the internet so that users having this software can decompress it.

2. DECOMPRESS THE FILE OR FOLDER

This is the reverse process of file compression. Here we can decompress the compressed file and get the original file.

3. VIEW FILES IN THE COMPRESSED FILE

Here we can view the list of files inside our compressed file. We can view the files before decompressing and decide to decompress or not.

4. SET EXTENSION

This is additional feature in our project. We can set our own extension to the compressed file.

5. SET ICON

We can specify the style of icon for the compressed file. Users will also be given a option to change the icon as per their preference.

APPLICATION AREAS

The application areas of file compression are

- File storage
- Distributed systems.

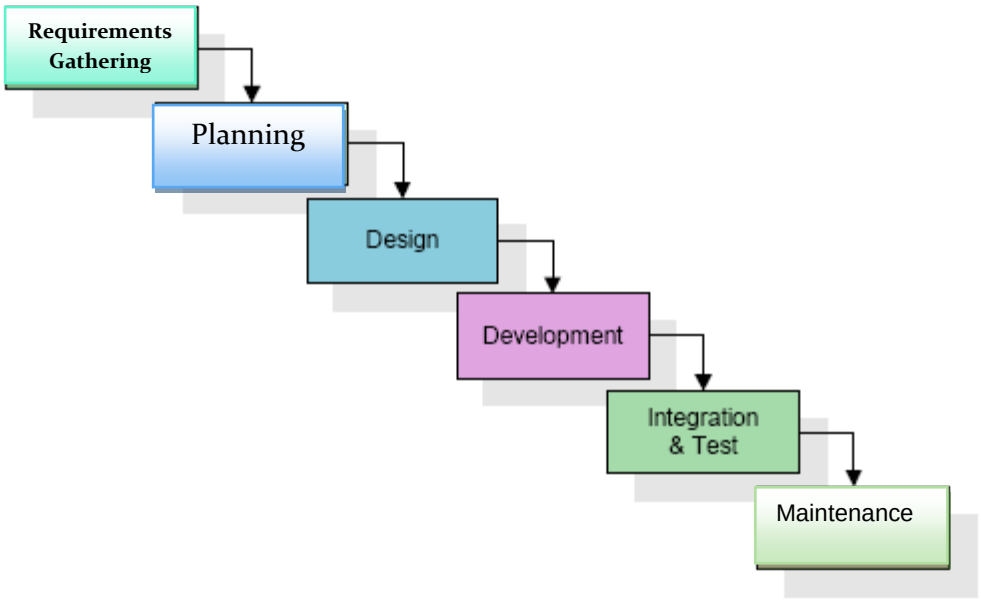
Chapter-4

System Development Life Cycle

SYSTEM DEVELOPMENT LIFE CYCLE

Acronym for SDLC is Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

The six stages of the SDLC are designed to build on one another, taking the outputs from the previous stage, adding additional effort, and producing results that leverage the previous effort and are directly traceable to the previous stages. This top-down approach is intended to result in a quality product that satisfies the original intentions of the customer.



STAGES IN SDLC

- Requirement Gathering
- Planning
- Design
- Coding
- Testing

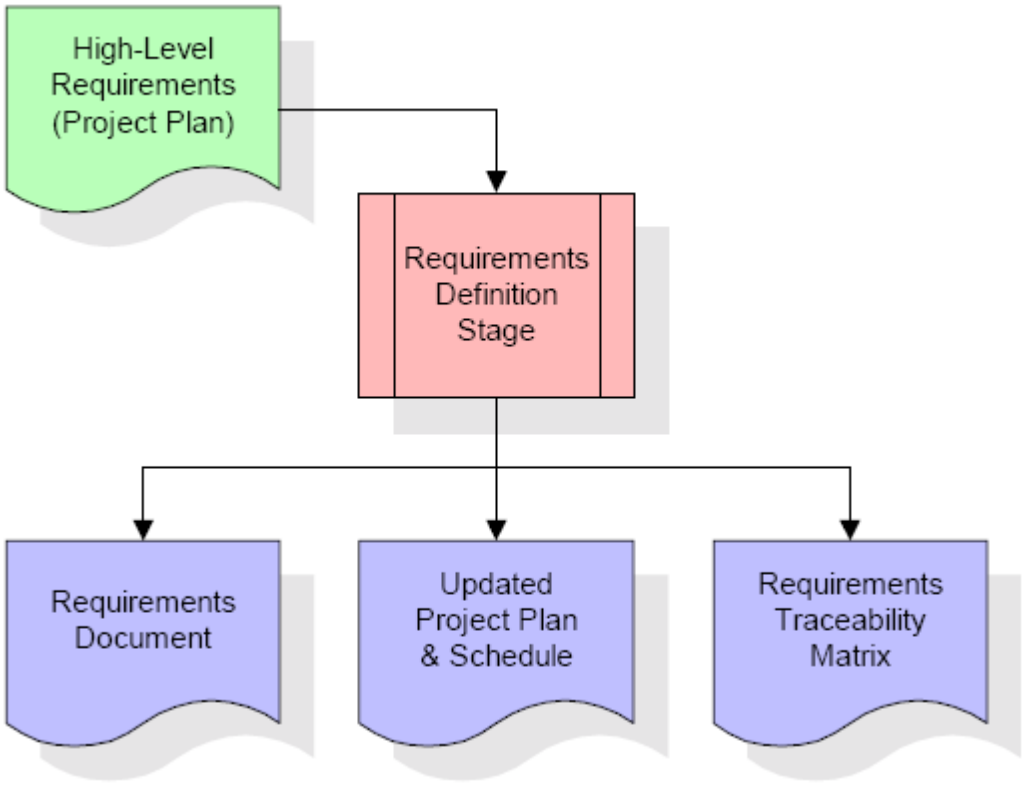
- Maintenance
- **REQUIREMENT GATHERING STAGE**

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements.

These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities.

Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.

These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are *not* included in the requirements document.



The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

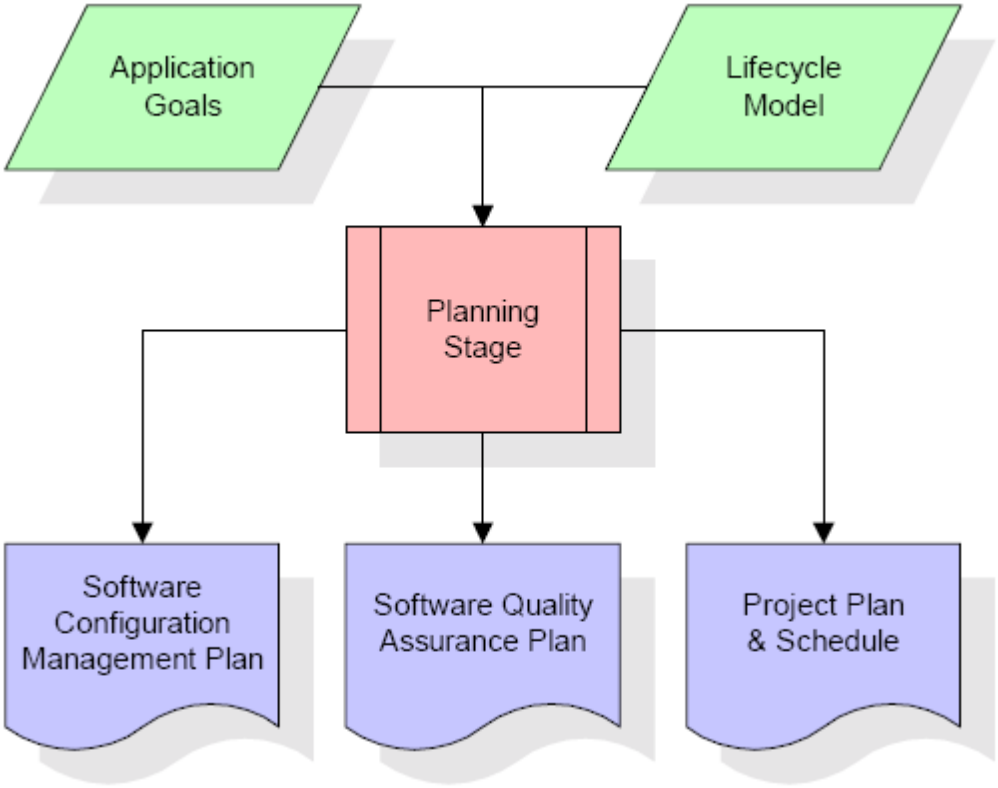
In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term requirements traceability.

The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

ANALYSIS OR PLANNING STAGE

The analysis stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical

approaches.



The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal

consists of a title and textual description, although additional information and references to external documents may be included.

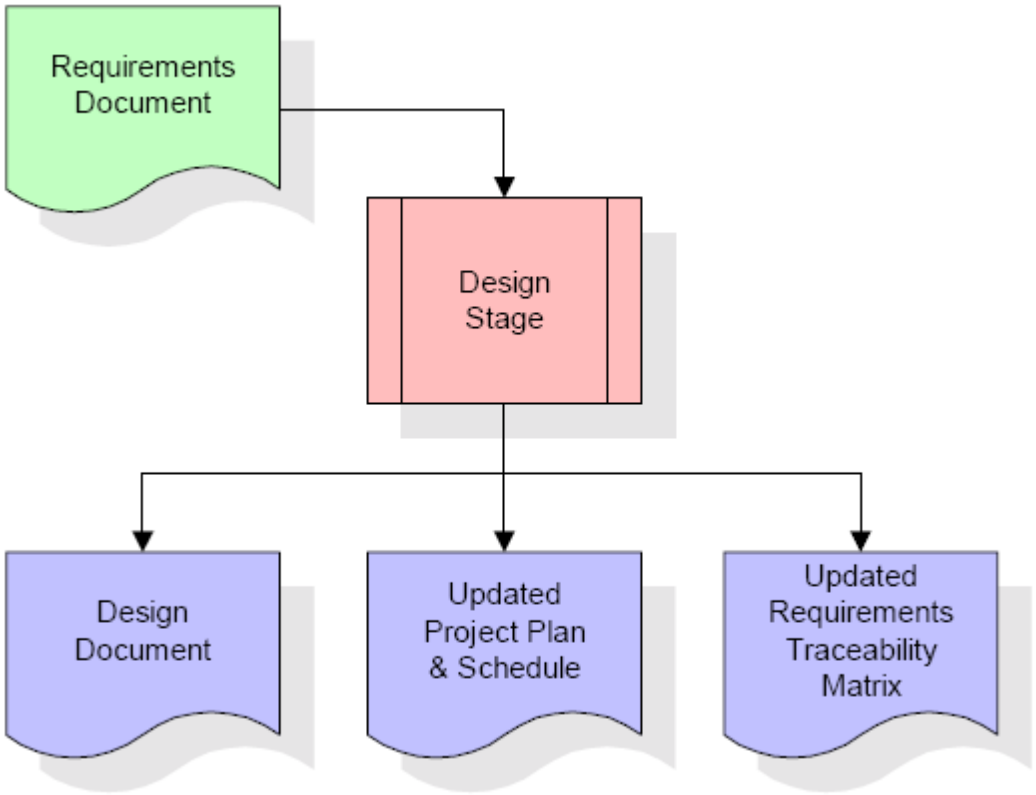
The outputs of the project analysis stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and highlevel estimates of effort for the out stages.

DESIGN STAGE

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts.

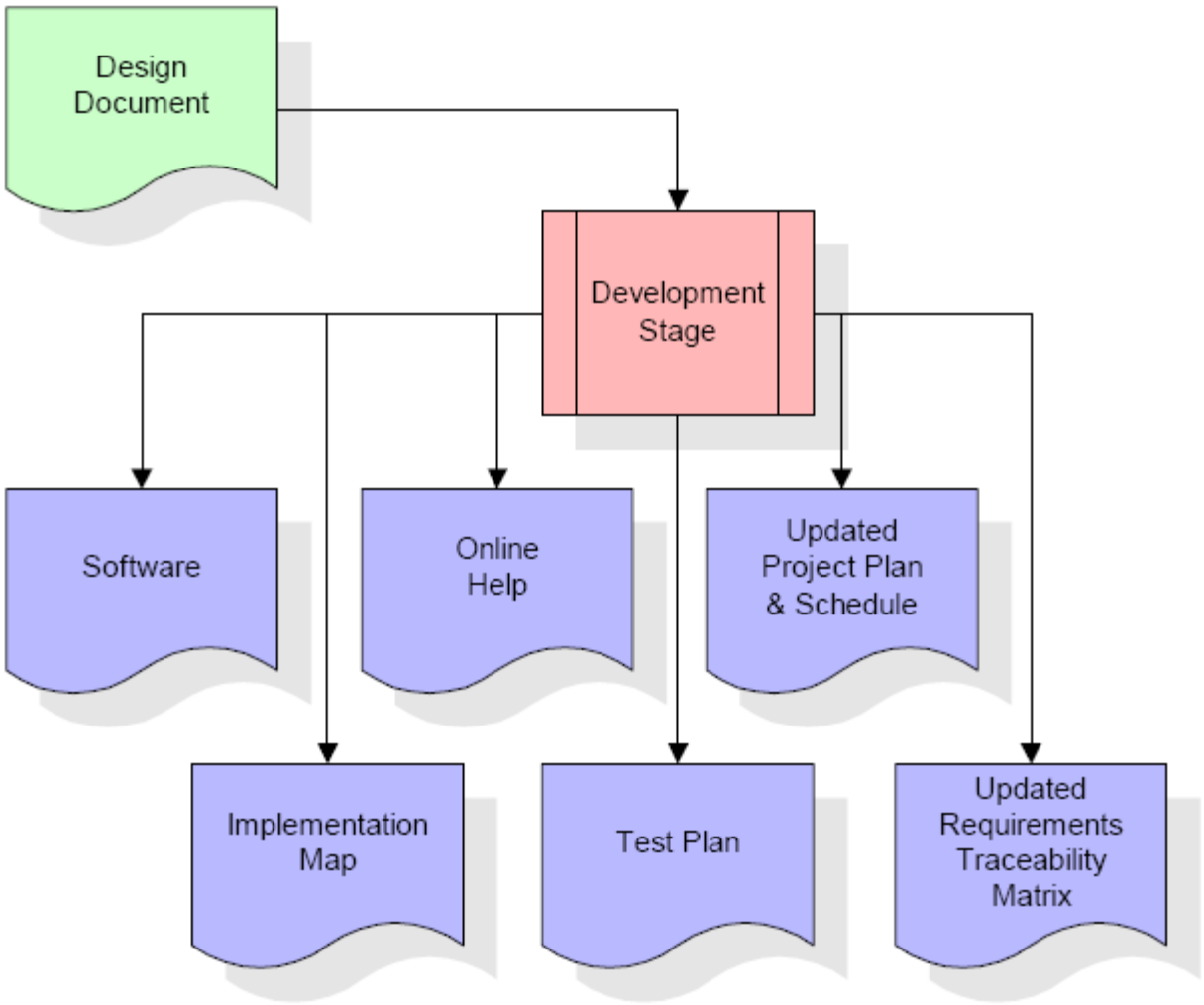
Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudocode, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.

When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.



CODING OR DEVELOPMENT STAGE

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.



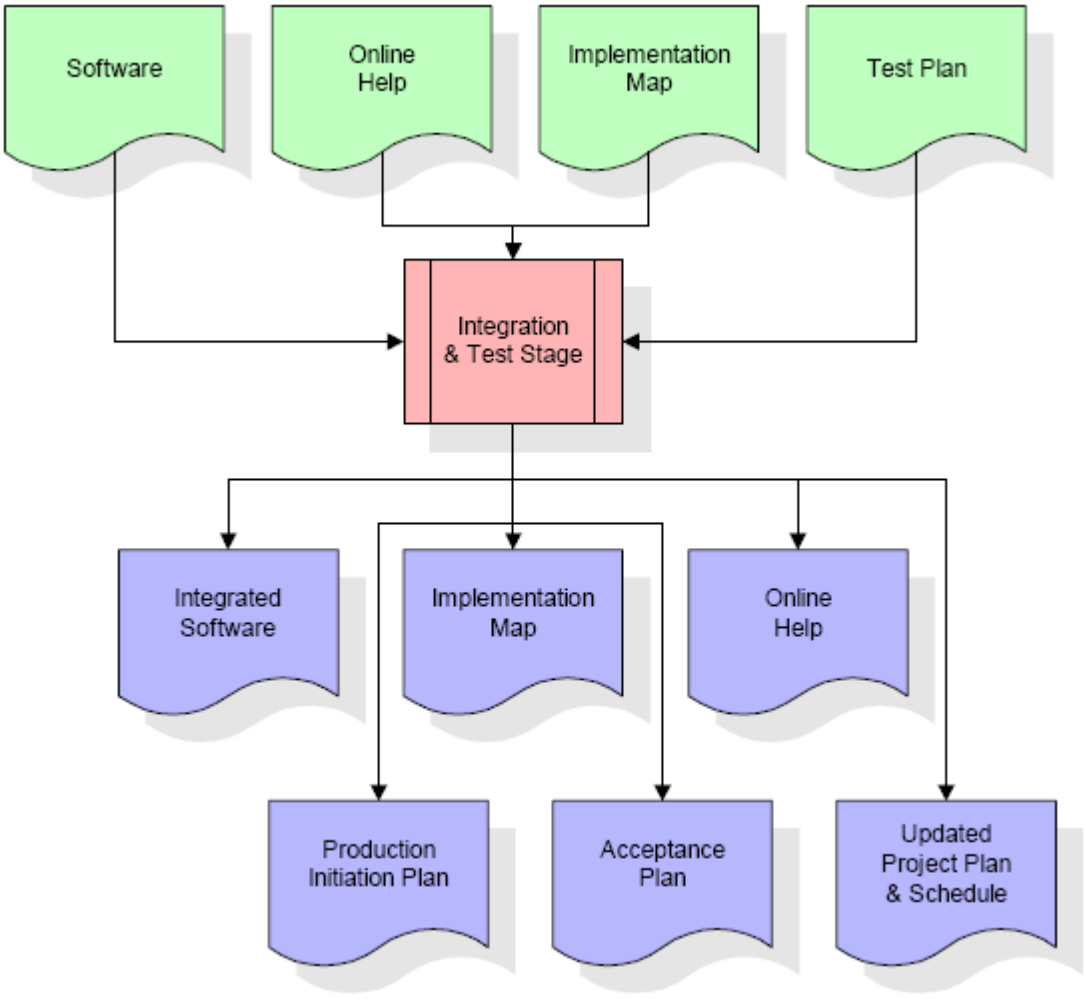
The RTM will be updated to show that each developed artifact is linked to a specific design element, and that each developed artifact has one or more corresponding test case items. At this point, the RTM is in its final configuration.

The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be

used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

TESTING OR INTEGRATION STAGE

During the integration and test stage, the software artifacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability.



During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.

The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

MAINTENANCE STAGE

A project is already developed completely and it is in production Issues are posted from production and support team from client analyses the issues and assigns the same to be fixed on offshore. So testing of such issues has to be done after the fixing of the issue according to the expectation and requirement specs provided by the client. In order to ensure effective testing - Tester need to describe the scenarios that were tested (when there are no test cases provided for the same).

Chapter-5

System Analysis

SYSTEM ANALYSIS

System analysis is a process of gathering and interpreting facts, diagnosing problems and the information to recommend improvements on the system. It is a problem solving activity that requires intensive communication between the system users and system developers. System analysis or study is an important phase of any system development process. The system is studied to the minutest detail and analyzed. The system analyst plays the role of the interrogator and dwells deep into the working of the present system. The system is viewed as a whole and the input to the system are identified. The outputs from the organizations are traced to the various processes. System analysis is concerned with becoming aware of the problem, identifying the relevant and decisional variables, analyzing and synthesizing the various factors and determining an optimal or at least a satisfactory solution or program of action.

A detailed study of the process must be made by various techniques like interviews, questionnaires etc. The data collected by these sources must be scrutinized to arrive to a conclusion. The conclusion is an understanding of how the system functions. This system is called the existing system. Now the existing system is subjected to close study and problem areas are identified. The designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for an endorsement by

the user. The proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user is satisfied with proposal.

Preliminary study is the process of gathering and interpreting facts, using the information for further studies on the system. Preliminary study is problem solving activity that requires intensive communication between the system users and system developers. It does various feasibility studies. In these studies a rough figure of the system activities can be obtained, from which the decision about the strategies to be followed for effective system study and analysis can be taken.

Here in the project SunZip, a detailed study of existing system is carried along with all the steps in system analysis. An idea for creating a better project was carried and the next steps were followed.

FEASIBILITY STUDY

An important outcome of the preliminary investigation is the determination that the system requested is feasible. Feasibility study is carried out to select the best system that meets the performance requirements.

Feasibility study is both necessary and prudent to evaluate the feasibility of the project at the earliest possible time. It involves preliminary investigation of the project and examines whether the designed system will be useful to the organization. Months or years of effort, thousand for millions of money and untold professional embarrassment can be averted if an in-conceived system

is recognized early in the definition phase.

The different types of feasibility are: Technical feasibility, Operational feasibility, Economical feasibility.

1. TECHNICAL FEASIBILITY

Technical Feasibility deals with the hardware as well as software requirements. Technology is not a constraint to type system development. We have to find out whether the necessary technology, the proposed equipments have the capacity to hold the data, which is used in the project, should be checked to carryout this technical feasibility.

The technical feasibility issues usually raised during the feasibility stage of investigation includes these

- This software is running in windows 2000 Operating System, which can be easily installed
- The hardware required is Pentium based server
- The system can be expanded

2. OPERATIONAL FEASIBILITY

This feasibility test asks if the system will work when it is developed and installed.

Operational feasibility in this project:

- The proposed system offers greater level of user-friendliness
- The proposed system produces best results and gives high

performance. It can be implemented easily .So this project is operationally feasible

3. ECONOMICAL FEASIBILITY

Economical Feasibility deals about the economical impact faced by the organization to implement a new system. Financial benefits must equal or exceed the costs. The cost of conducting a full system, including software and hardware cost for the class of application being considered should be evaluated.

Economic Feasibility in this project:

- The cost to conduct a full system investigation is possible
- There is no additional manpower requirement
- There is no additional cost involved in maintaining the proposed system

5.1. EXISTING SYSTEM

Existing system refers to the system that is being followed till now. The main disadvantage of this system is that the users depend on third party software's like winzip, winrar, Stuff etc.

The existing system requires more computational time, more manual calculations, and the complexity involved in Selection of features is high. The other disadvantages are lack of security of data, Deficiency of Data accuracy, Time consuming etc.

To avoid all these limitations and make the working more accurately the system needs to be computerized.

5.2. PROBLEM STATEMENT

- Lack of security of data
- Deficiency of Data accuracy
- Time consuming
- The users depend on third party software's like winzip, winrar etc.

To avoid all these limitations and make the working more accurately the system needs to be computerized.

5.3. PROPOSED SYSTEM

The aim of proposed system is to develop a system of improved facilities. The proposed system can overcome all the limitations of the existing system. The system provides data accuracy and save disc space. The existing system has several disadvantages and many more difficulties to work well. The proposed system tries to eliminate or reduce these difficulties up to some extent. The proposed system is file/folder compression or decompression based on the Huffman algorithm and GZip algorithm. The proposed system will help the user to consume time. The proposed system helps the user to work user friendly and he can easily do the file compression process without time lagging. The system is very simple in design and to implement. The system requires very low system resources and the system will work in almost all configurations. It has got following features Ensure data

accuracy, minimize manual data entry, minimum time needed for the various processing, greater efficiency, better service.

5.4. USES AND ADVANTAGES

The system is very simple in design and to implement. The system requires very low system resources and the system will work in almost all configurations. It has got following features

- Ensure data accuracy and Save disk space
- Minimum time needed for the file compression
- Greater efficiency and Better Service
- Protection from virus and Easy to send via E-mail
- Maximum Compression rate is 2 GB
- The user need not depend on third party software's like winzip, winrar etc.

Chapter-6

System Requirements

SYSTEM REQUIREMENTS

6.1. HARDWARE REQUIREMENTS

- Processor : Intel Pentium IV 2.4 GHZ or above
- Clock speed : 500 MHZ
- System bus : 32 bits
- RAM : 256MB of RAM
- HDD : 40 GB or higher
- Monitor : SVGA COLOR
- Keyboard : 108 keys
- Mouse : 2 button mouse

6.2. SOFTWARE REQUIREMENTS

- Operating System : MS WINDOWS XP SP2
- Front End : Core Java
- Back End : My SQL

Chapter-7

System Environment

SYSTEM ENVIRONMENT

7.1. TECHNOLOGIES USED

This project is implemented using Java. Java goes back to 1991 when a group of sun engineers led by James Gosling , wanted to design a small computer language that could be used for consumer devices and named it as Green Project. Their idea was to develop a portable language that could generate intermediate code for virtual machines. This intermediate code then can be used on any machines that has the correct interpreter.

Java is a programming language that lets us to do almost anything we can do with traditional programming language for distributed applications. It is platform in-dependent and having a lot of networking features included within it. A java program can run equally well on any architecture that has a java interpreter.

FEATURES OF JAVA

1. ENCAPSULATION

Data Encapsulation is one of the most sticking features of OOP's. Encapsulation is the wrapping up of data and function into single unit called class. The wrapped defines the behaviour and protects the code and data from being arbitrarily accessed by the outside world and only those function which are wrapped in the class can access it. This type of insulation of data from direct access by the program is called data hiding.

2. INHERITANCE

Inheritance is the process by which objects a class can acquire the properties of objects of another class i.e. In OOPs the concept of inheritance provides idea of reusability providing the means of adding additional features to an existing class without

modifying it. This is possible by deriving a new class from the existing on thus the newly created class will have the combined features of both the parent and the child classes.

3. OBJECT ORIENTED

Almost everything in java is a clear, a method or an object. Only the most basic primitive operative and data types are at a sub-class level.

4. DATA ABSTRACTION

Data Abstraction is an act of representing essential features without including the background details and explanation.

5. PLATFORM INDEPENDENT

Java programs are compiled with a byte code format that can be read and run by interpreters on many platforms including Windows 95, Windows NT and later.

6. MULTI-THREADING

Java is inherently multi-threaded. A single java program can make many different things processing independently and continuously.

7. HIGH PERFORMANCE

Java can be compiled on the fly with a Just-in-time compiler (JIT) to

code that rivals C++ in speed.

8. SAFE

Java code can be executed in an environment that prohibits it from viruses, deleting or modifying files or otherwise performing data destroying and computer crashing operation.

9. SIMPLE

Java has the bare bones functionally needed to implement its rich feature set.

COMPONENTS

Java has several in-built components:

- Javac : Compiler for java programs that could generate byte codes
- Java : Interpreter to read and execute java byte codes.
- Javap : To disassemble and debug the java bytecodes.
- Javadoc : Document generator.
- Javah : To write and link native codes with java programs.

7.2. MODELING TOOL

1. NETBEANS

NetBeans refers to both a platform framework for Java desktop applications, and an integrated development environment (IDE) for developing

with Java, JavaScript, PHP, Python, Ruby, Groovy, C, C++, Scala, Clojure and much more (for a complete overview visit the website of netbeans).

The NetBeans IDE is written in Java and runs everywhere where a JVM is installed, including Windows, Mac OS, Linux, and Solaris. A JDK is required for Java development functionality, but is not required for development in other programming languages.

The NetBeans Platform allows applications to be developed from a set of modular software components called *modules*. Applications based on the NetBeans platform (including the NetBeans IDE) can be extended by third party developers.

NETBEANS PLATFORM

The NetBeans Platform is a reusable framework for simplifying the development of Java Swing desktop applications. The NetBeans IDE bundle for Java SE contains what is needed to start developing NetBeans plugins and NetBeans Platform based applications; no additional SDK is required.

Applications can install modules dynamically. Any application can include the Update Center module to allow users of the application to download digitally-signed upgrades and new features directly into the running application. Reinstalling an upgrade or a new release does not force users to download the entire application again.

The platform offers reusable services common to desktop applications, allowing developers to focus on the logic specific to their application. Among the features of the platform are:

- User interface management (e.g. menus and toolbars)
- User settings management
- Storage management (saving and loading any kind of data)
- Window management
- Wizard framework (supports step-by-step dialogs)
- NetBeans Visual Library

NETBEANS IDE

The NetBeans IDE is an open-source integrated development environment. NetBeans IDE supports development of all Java application types (Java SE including JavaFX, (Java ME, web, EJB and mobile applications) out of the box. Among other features are an Ant-based project system, Maven support, refactorings, version control (supporting CVS, Subversion, Mercurial and Clearcase).

2. ECLIPSE

Eclipse is a multi-language software development environment comprising an integrated development environment (IDE) and an extensible plug-in system. It is written primarily in Java and can be used to develop applications in Java and, by means of various plug-ins, other languages

including C, C++, COBOL, Python, Perl and PHP. The IDE is often called Eclipse ADT for Ada, Eclipse CDT for C/C++, Eclipse JDT for Java and Eclipse PDT for PHP.

The initial codebase originated from VisualAge.[1] In its default form it is meant for Java developers, consisting of the Java Development Tools (JDT). Users can extend its capabilities by installing plug-ins written for the Eclipse software framework, such as development toolkits for other programming languages, and can write and contribute their own plug-in modules.

Chapter-8

Detailed System Design

DETAILED SYSTEM DESIGN

SYSTEM DESIGN

System Design is the most creative and challenging phase in the system life cycle. Design is the first step into the development phase for any engineered product or system. Design is a creative process. A good design is the key to effective system. System design is a solution *how to approach* the creation of a new system. System design transforms a logic representation of what is required to do into the physical specification. The specification is converted into physical reality during development.

LOGICAL DESIGN

The logical flow of a system and define the boundaries of a system. It includes the following steps:

- Reviews the current physical system – its data flows, file content, volumes, frequencies etc
- Prepares output specifications – that is, determines the format, content and frequency of reports
- Prepares input specifications – format, content and most of the input functions
- Prepares edit, security and control specifications
- Specifies the implementation plan
- Prepares a logical design walk through of the information flow, output, input, controls and implementation plan

- Reviews benefits, costs, target dates and system constraints

PHYSICAL DESIGN

Physical system produces the working systems by define the design specifications that tell the programmers exactly what the candidate system must do. It includes the following steps.

- Design the physical system
- Specify input and output media
- Design the database and specify backup procedures
- Design physical information flow through the system and a physical design walk through
- Plan system implementation
- Prepare a conversion schedule and target date
- Determine training procedures, courses and timetable
- Devise a test and implementation plan and specify any new hardware/software
- Update benefits , costs , conversion date and system constraints
- Design/Specification activities

- Concept formulation
- Problem understanding
- High level requirements proposals
- Feasibility study
- Requirements engineering

- Architectural design

INPUT DESIGN

Input Design deals with what data should be given as input, how the data should be arranged or code, the dialog to guide the operating personnel in providing input, methods for preparing input validations and steps to follow when error occur. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow.

In this project, the input design consists of a log in screen, tab for compression/ decompression, source and destination browsing button, a menu list for selecting the algorithm, Compress/Decompress option, compress/decompress button.

OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. The objective of output design is to convey information about past activities, current status or projections of the future, signal important events, opportunities, problems, or warnings, trigger an action, confirm an action etc. Efficient, intelligible output design should improve the system's relationship with the user and helps in decisions making. In output design the emphasis is on displaying the output on a CRT screen in a predefined format. The primary consideration in design of output is the information requirement and objectives of the end users. The major formation of the output is to convey the information and so its layout and design need a careful consideration.

There is a output display screen for showing the compressed/ decompressed file or folder details(Original file size, Compressed/Decompressed file size, Distinct characters)

DATA FLOW DIAGRAM

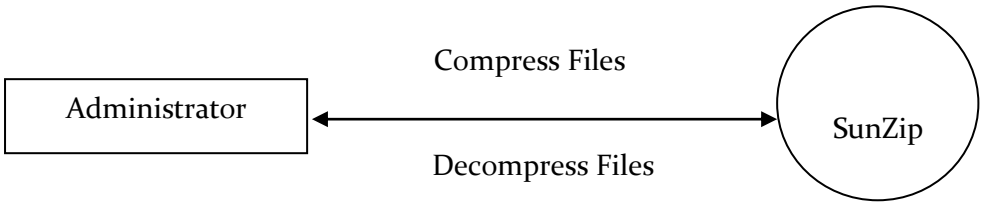
Data flow oriented techniques advocate that the major data items handled by a system must be first identified and then the processing required on these data items to produce the desired outputs should be determined. The DFD (also called as bubble chart) is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on these data, and the output generated by the

system. It was introduced by De Macro (1978), Gane and Sarson (1979).The primitive symbols used for constructing DFD's are:

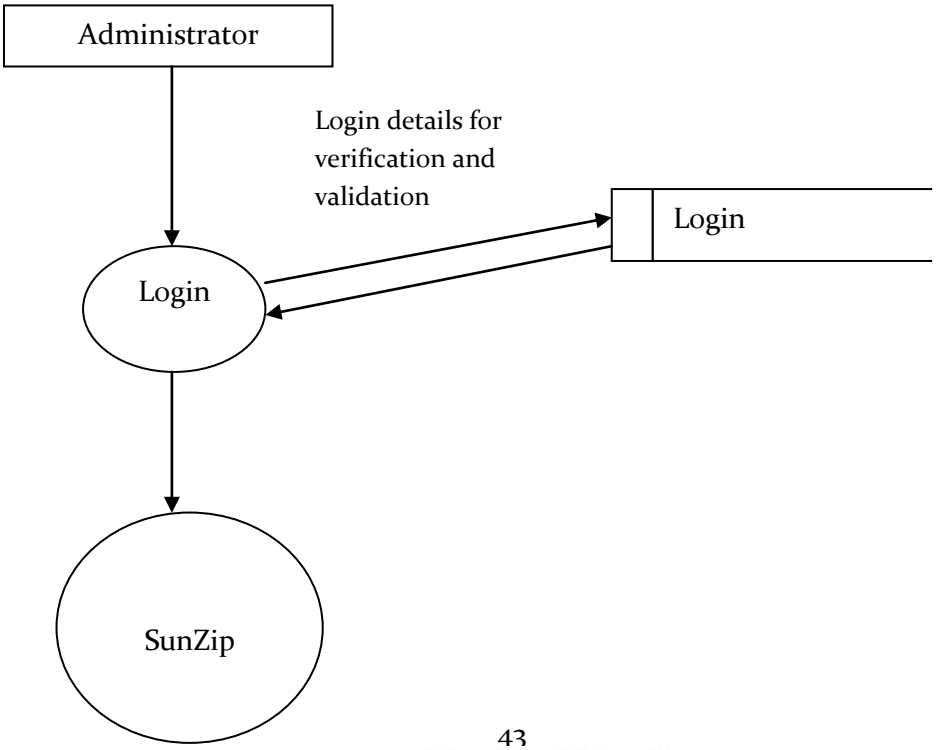
SYMBOLS USED IN DFD

- A *circle* represents a process.
- A *rectangle* represents external entity
- A *square* defines a source or destination of the system data.
- An *arrow* identifies dataflow
- A *double line* with one end closed indicates data store

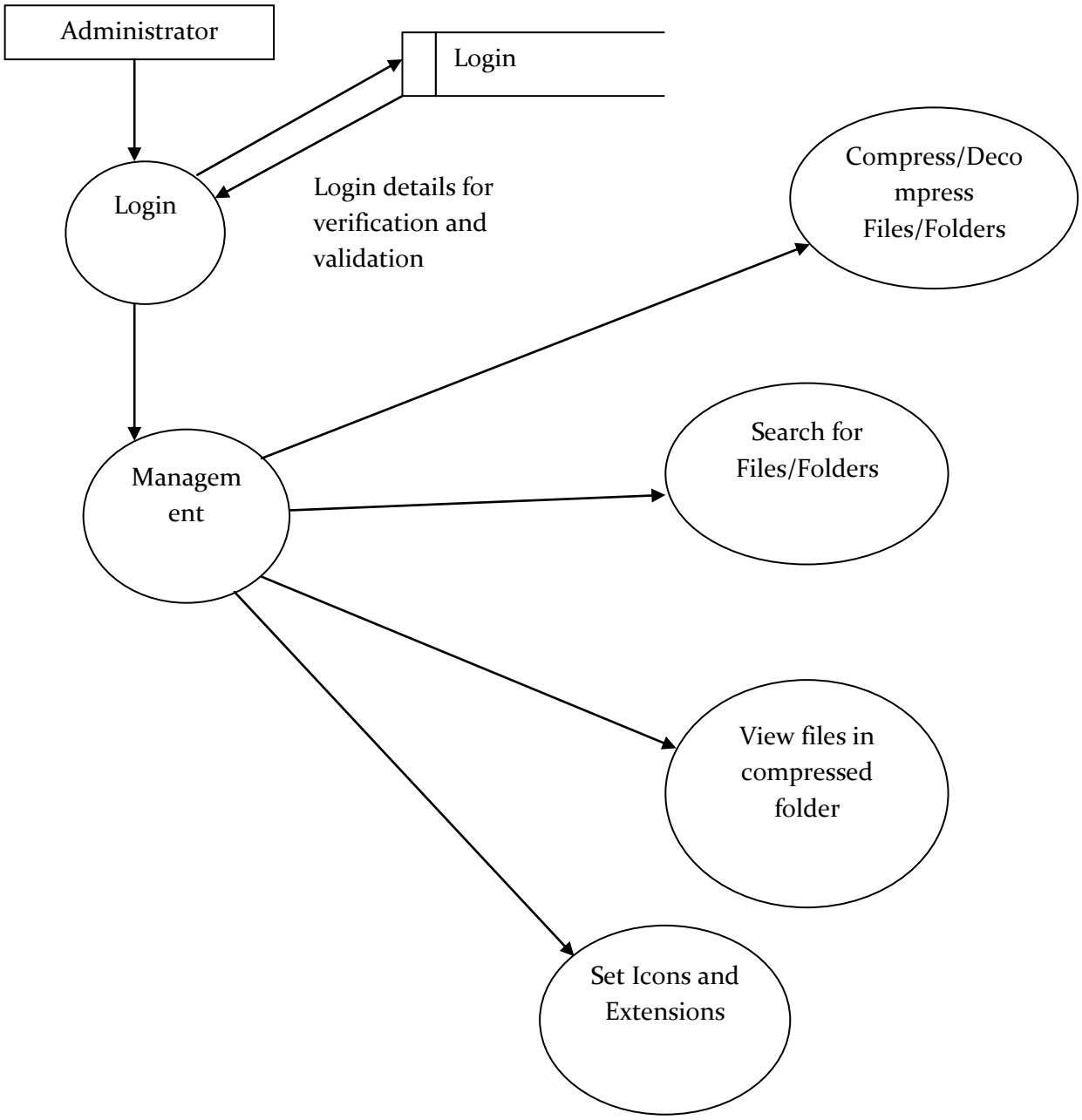
LEVE 0 DFD ADMINISTRATOR



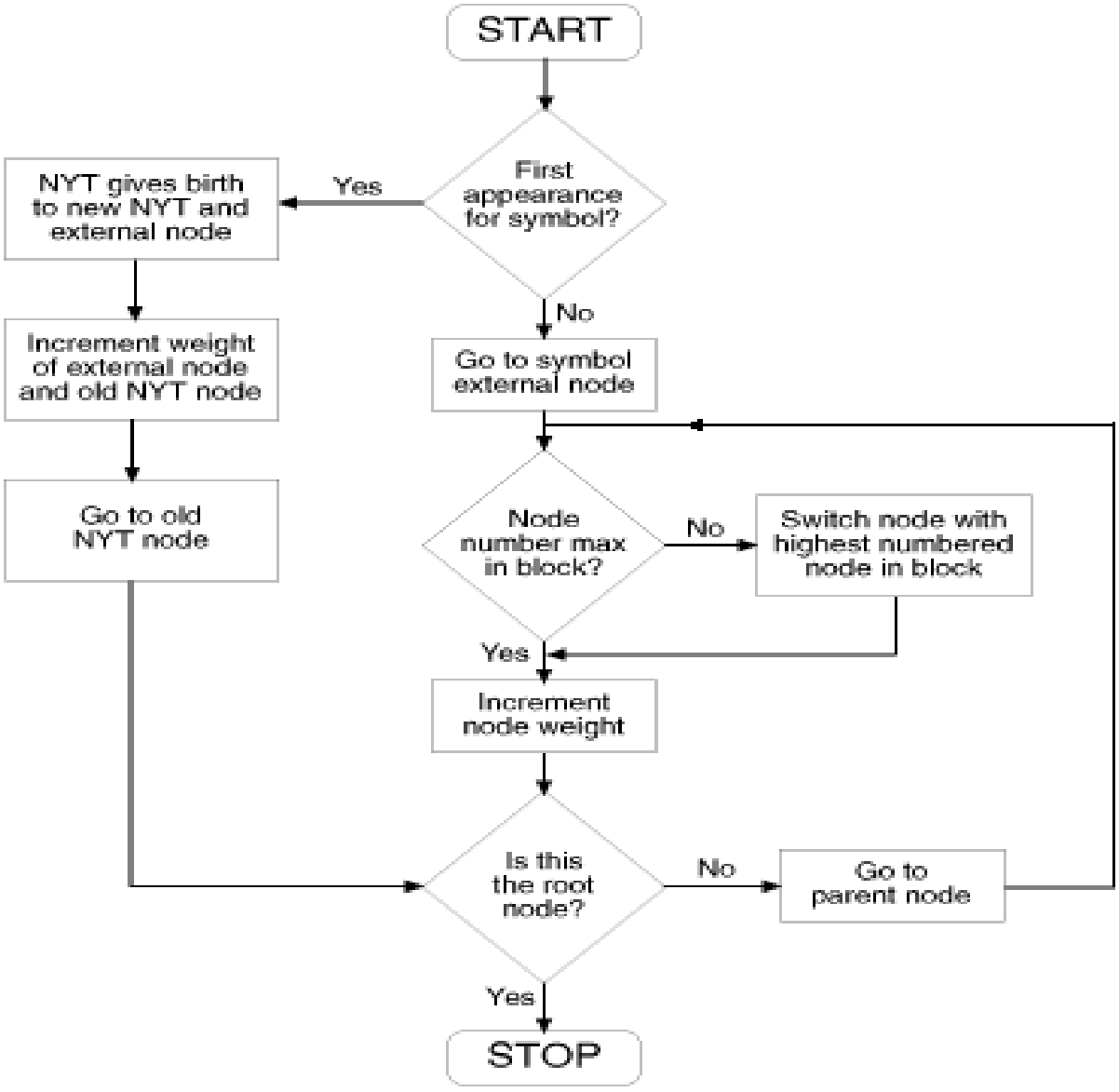
LEVEL 1 DFD ADMINISTRATOR



LEVE 2 DFD ADMINISTRATOR



FLOW CHART



8.1. ARCHITECTURE

TWO-TIER ARCHITECTURE

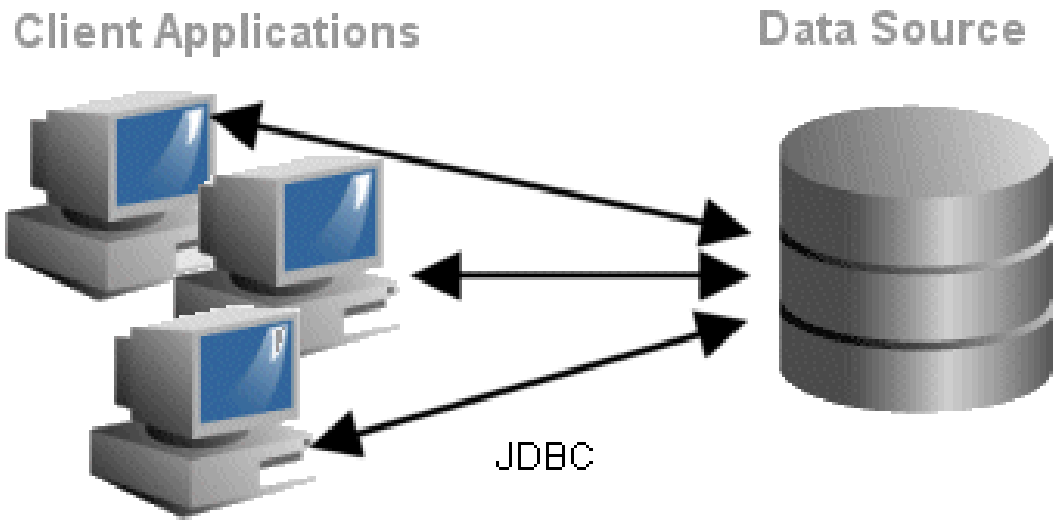
A two-tier application generally includes a Java client that connects directly to the database through TopLink. The two-tier architecture is most common in complex user interfaces with limited deployment. The database session provides TopLink support for two-tier applications.

Although the two-tier architecture is the simplest TopLink application pattern, it is also the most restrictive, because each client application requires its own session. As a result, two-tier applications do not scale as easily as other architectures.

Two-tier applications are often implemented as user interfaces that directly access the database. They can also be non-interface processing engines.

The following are key elements of efficient two-tier (client-server) architecture with TopLink:

- Minimal dedicated connections from the client to the database
- An isolated object cache



EXAMPLE IMPLEMENTATIONS

- An example of a two-tier architecture implementation is a Java user interface (Swing/AWT) and batch data processing.

ADVANTAGE

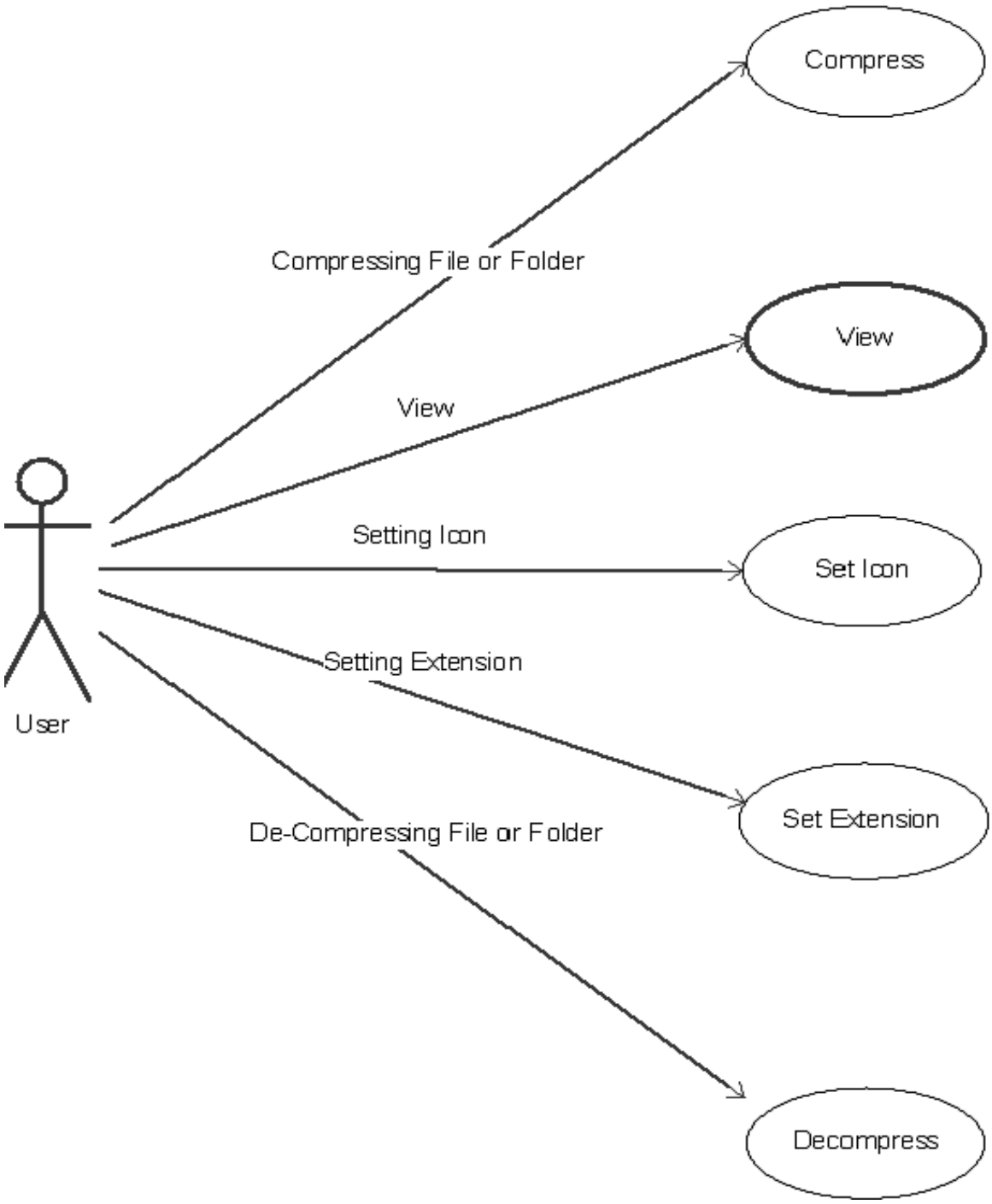
- The advantage of the two-tier design is its simplicity. The TopLink database session that builds the two-tier architecture provides all the TopLink features in a single session type, thereby making the two-tier architecture simple to build and use.

DISADVANTAGES

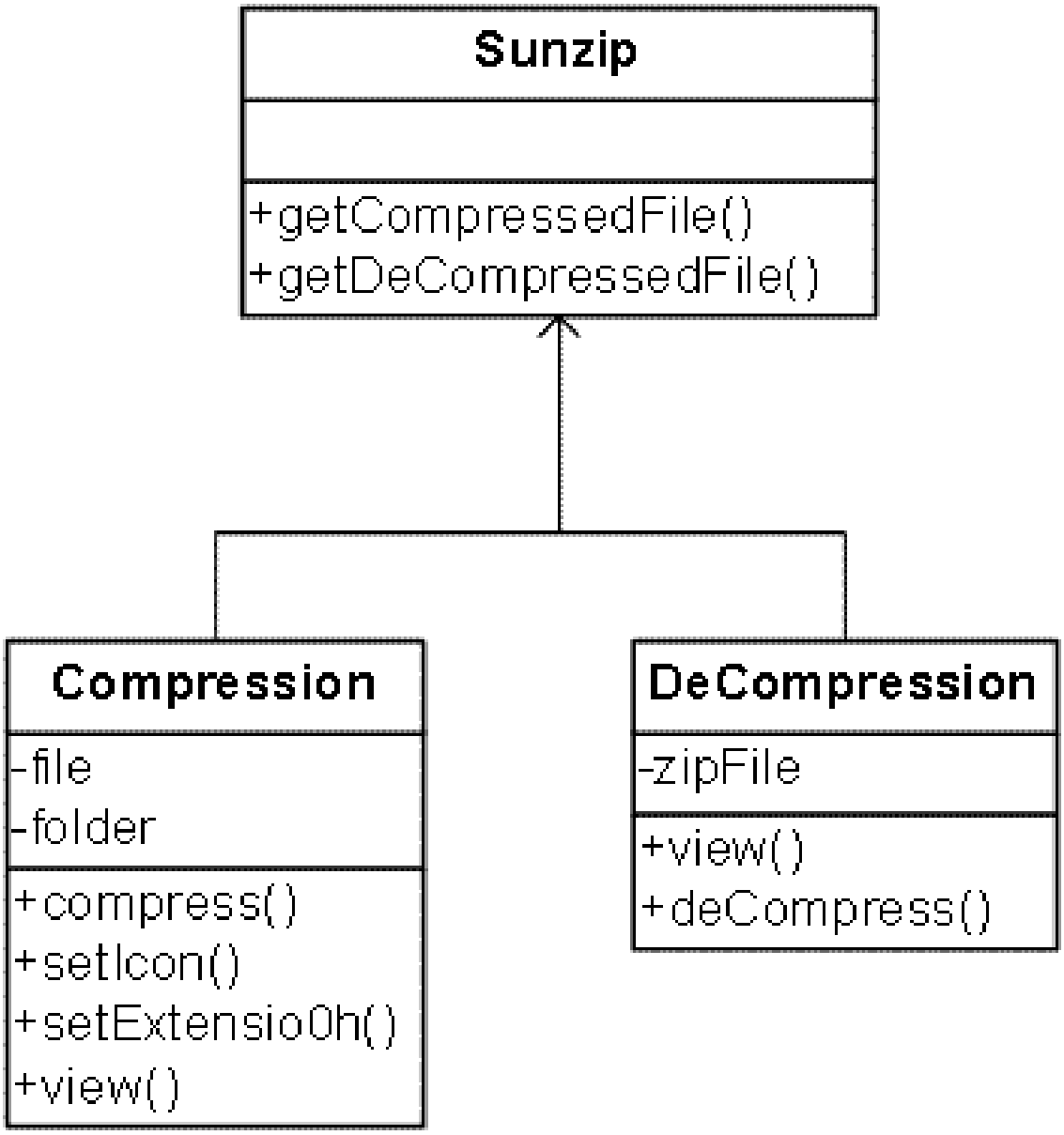
- The most important limitation of the two-tier architecture is that it is not scalable, because each client requires its own database session.

8.2. UML DIAGRAMS

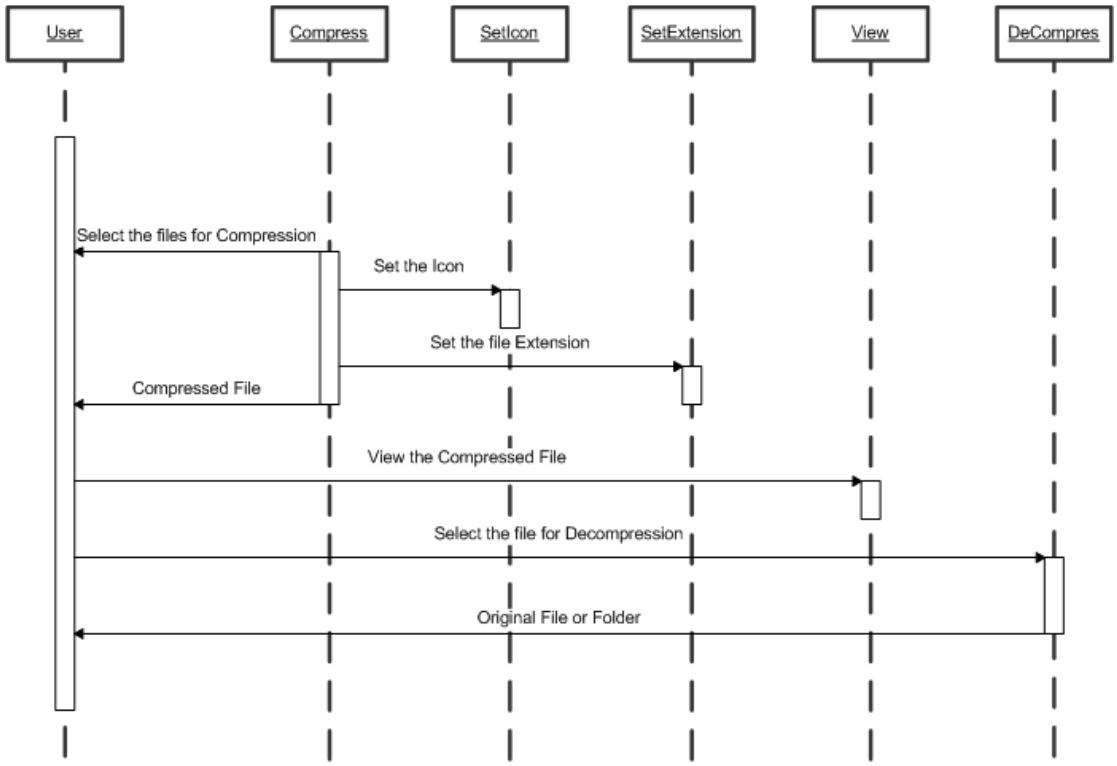
USECASE DIAGRAM



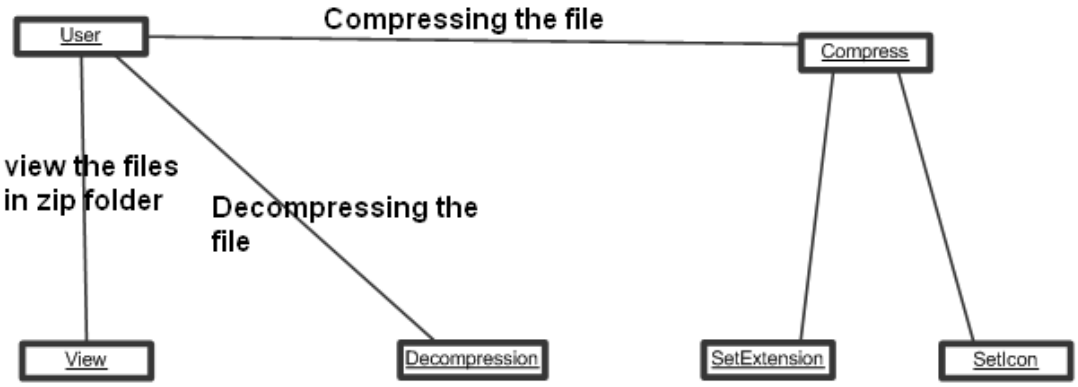
CLASS DIAGRAM



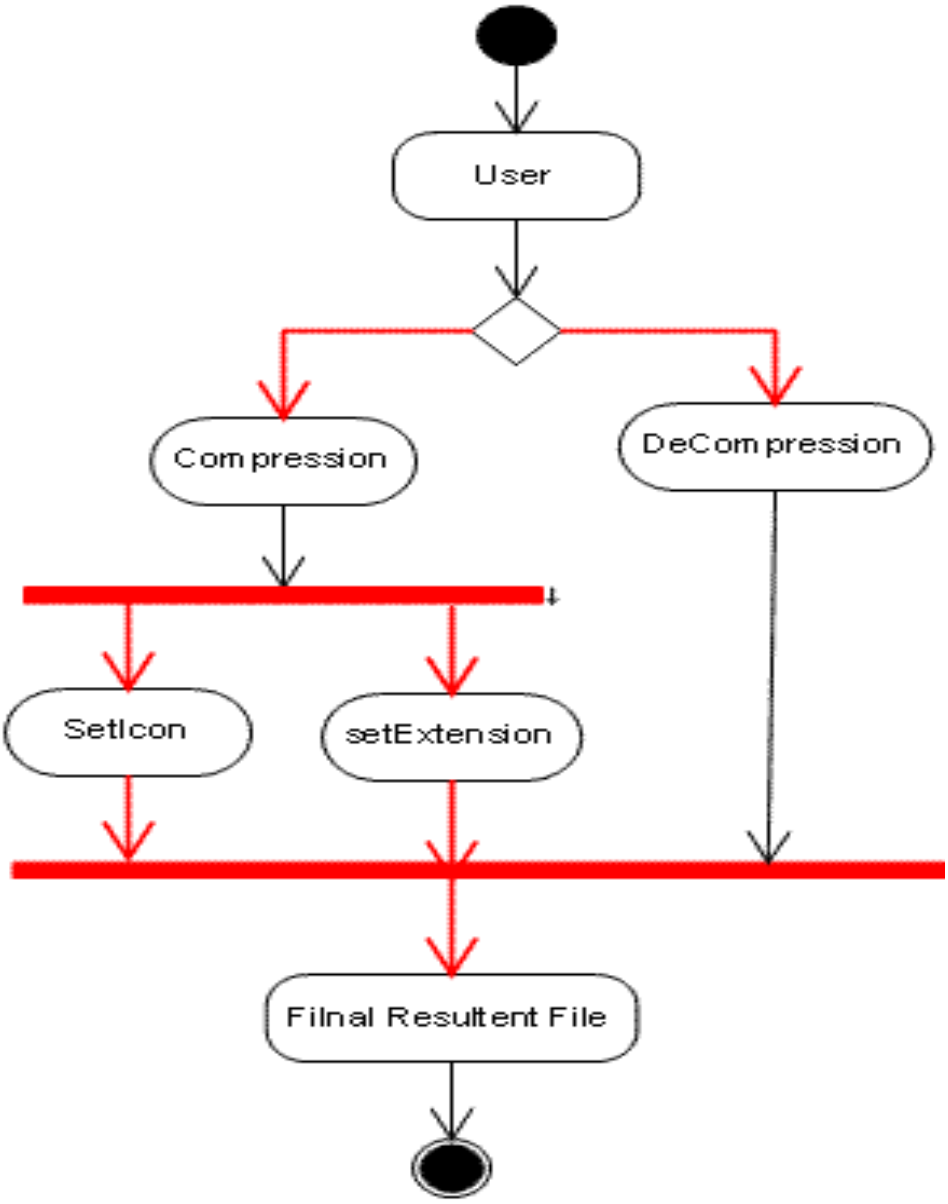
SEQUENCE DIAGRAM



COLLABORATION DIAGRAM



ACTIVITY DIAGRAM



Chapter-9

System Coding and Implementation

SYSTEM CODING AND IMPLEMENTATION

9.1. SAMPLE CODE

//1: Eve.java

```
import javax.swing.*;
public class Eve {
    private static void  createAndShowGUI(){
        EveFrame mainframe = new EveFrame();
        mainframe.setVisible(true);
    }
    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                createAndShowGUI();
            }
        });
    }
}
```

//2: EveAbout.java

```
import java.awt.*;
import javax.swing.*;
public class EveAbout extends JPanel {
    JLabel splashImage;
```

```
    EveAbout(JFrame parent,boolean isDblBuf){
        super(isDblBuf);
        setLayout(new BorderLayout());
    splashImage=new JLabel(new
ImageIcon(getClass().getResource("/images/eve_about.png")));
        add(splashImage, BorderLayout.CENTER);
        splashImage.setBorder(BorderFactory.createLineBorder(new Color(75,
75, 75)));
    }
}
```

//3: EveComp.java

```
import java.io.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class EveComp extends JPanel implements
ActionListener,ItemListener,EveGuiConstants{
    final String REFRESH = "\t   ----- Status -----";
    private JTextField txtSource,txtDestination ;
    private JTextArea txtStatus;
    private JButton btnSource,btnDestination,btnAction,btnClear;
    private JComboBox cmbAlgorithms;
    private JRadioButton rbtnCompress,rbtnDecompress;
```

```
private ButtonGroup btnGroup;
private JFileChooser fc = new JFileChooser();
private int algoSelected = COMP_HUFFMAN;
private JFrame owner;
EveComp(JFrame parent,boolean isDblBuf){
    super(isDblBuf);
    owner = parent;
    buildBasicPanel();
    rbtnCompress.addItemListener(this);
    rbtnDecompress.addItemListener(this);
    btnSource.addActionListener(this);
    btnDestination.addActionListener(this);
    btnClear.addActionListener(this);
    btnAction.addActionListener(this);
    cmbAlgorithms.addActionListener(this);
    txtStatus.setText(REFRESH);
}
void buildBasicPanel(){
    JPanel panel = this;
    GridBagLayout gridbag = new GridBagLayout();
    GridBagConstraints constraints = new GridBagConstraints();
    JLabel lblTest;
    panel.setLayout(gridbag);
    constraints.insets = new Insets(3,3,3,3);
```

```
buildConstraints(constraints,0,0,1,1,5,33);
lblTest = new JLabel("Source : ");
gridbag.setConstraints(lblTest,constraints) ;
panel.add(lblTest);
buildConstraints(constraints,1,0,2,1,95,33);
txtSource = new JTextField(250);
gridbag.setConstraints(txtSource,constraints) ;
panel.add(txtSource);
buildConstraints(constraints,3,0,1,1,0,0);
btnSource = new JButton("...");
gridbag.setConstraints(btnSource,constraints) ;
panel.add(btnSource);
buildConstraints(constraints,0,1,1,1,0,33);
lblTest = new JLabel("Destination : ");
gridbag.setConstraints(lblTest,constraints) ;
panel.add(lblTest);
buildConstraints(constraints,1,1,2,1,95,33);
txtDestination = new JTextField(250);
gridbag.setConstraints(txtDestination,constraints) ;
panel.add(txtDestination);
buildConstraints(constraints,3,1,1,1,0,0);
btnDestination = new JButton("...");
gridbag.setConstraints(btnDestination,constraints) ;
panel.add(btnDestination);
```

```
buildConstraints(constraints,0,2,1,1,0,33);
lblTest = new JLabel("Algorithms : ");
gridbag.setConstraints(lblTest,constraints) ;
panel.add(lblTest);
buildConstraints(constraints,1,2,3,1,0,0);
cmbAlgorithms = new JComboBox(algorithmNamesArray);
gridbag.setConstraints(cmbAlgorithms,constraints) ;
panel.add(cmbAlgorithms);
buildConstraints(constraints,0,3,1,1,0,33);
lblTest = new JLabel("Procedure : ");
gridbag.setConstraints(lblTest,constraints) ;
panel.add(lblTest);
constraints.anchor = GridBagConstraints.CENTER;       //Center
buildConstraints(constraints,1,3,1,1,0,33);
rbtnCompress = new JRadioButton(" Compress ",true);
gridbag.setConstraints(rbtnCompress ,constraints) ;
panel.add(rbtnCompress );
buildConstraints(constraints,2,3,1,1,0,33);
rbtnDecompress = new JRadioButton("Decompress");
//constraints.fill = GridBagConstraints.NONE;
gridbag.setConstraints(rbtnDecompress,constraints) ;
panel.add(rbtnDecompress);
buildConstraints(constraints,0,4,1,1,0,0);
lblTest = new JLabel("Status : ");
```

```
gridbag.setConstraints(lblTest,constraints) ;
panel.add(lblTest);
buildConstraints(constraints,1,4,3,2,0,200);
constraints.fill = GridBagConstraints.BOTH;
txtStatus = new JTextArea(5,20);
txtStatus.setMargin(new Insets(5,5,5,5));
JScrollPane logScrollPane = new JScrollPane(txtStatus);
logScrollPane.setBorder(BorderFactory.createLineBorder(Color.black));
gridbag.setConstraints(logScrollPane,constraints) ;
panel.add(logScrollPane );
buildConstraints(constraints,1,6,1,1,40,33);
btnAction = new JButton(" Compress ");
gridbag.setConstraints(btnAction ,constraints) ;
panel.add(btnAction );
buildConstraints(constraints,2,6,1,1,40,33);
//constraints.fill = GridBagConstraints.NONE;
btnClear = new JButton("Clear");
gridbag.setConstraints(btnClear,constraints) ;
panel.add(btnClear);
txtSource.setEditable(false);
txtStatus.setEditable(false);
txtDestination.setEditable(false);
btnGroup = new ButtonGroup();
btnGroup.add(rbtnCompress);
```

```
        btnGroup.add(rbtnDecompress);
    }
    void log(String stuff){
        if(txtStatus.getDocument().getLength() > 3000)
txtStatus.setText(REFRESH);
            txtStatus.append("\n" + stuff);
        txtStatus.setCaretPosition(txtStatus.getDocument().getLength());
    }
    void buildConstraints(GridBagConstraints gbc, int gx, int gy,int gw, int gh,
int wx, int wy) {
        gbc.fill = GridBagConstraints.HORIZONTAL;
        gbc.gridx = gx;
        gbc.gridy = gy;
        gbc.gridwidth = gw;
        gbc.gridheight = gh;
        gbc.weightx = wx;
        gbc.weighty = wy;
    }
    public void actionPerformed(ActionEvent e) {
        Object obj = e.getSource();
        if(obj instanceof JButton){
            if(obj == btnSource){
            int returnVal = fc.showOpenDialog(this);
            if (returnVal == JFileChooser.APPROVE_OPTION) {
```

```
            txtSource.setText("");
            txtDestination.setText("");
            File file = fc.getSelectedFile();
            if(file.exists()){
                log("Source : [" +  file.getName() + "]");
                txtSource.setText(file.getPath());
                suggestDestination();
            }
            }
        }else if(obj == btnDestination){
            int returnVal = fc.showSaveDialog(this);
            if (returnVal == JFileChooser.APPROVE_OPTION) {
                txtDestination.setText("");
                File file = fc.getSelectedFile();
                log("Destination : [" + file.getName() + "]");
                txtDestination.setText(file.getPath());
            }
        }else if(obj == btnClear){
            clearDetails();
            rbtnCompress.setSelected(true);
        }else if(obj == btnAction){
            if(txtSource.getText().length() == 0 ||
txtDestination.getText().length() == 0){
                log("Fill all Fields!");
```

```
                return;
            }
            EveWorkingDlg dlg = new EveWorkingDlg (owner);
            int mode = (rbtnCompress.isSelected())? COMPRESS :
DECOMPRESS;

dlg.doWork(txtSource.getText(),txtDestination.getText(),mode,algoSelected);
            log(dlg.getSummary());
          }
       }//JButton??
       else if(obj instanceof JComboBox){
          log("Algorithm Selected : [" + cmbAlgorithms.getSelectedItem() + "]");
          algoSelected = cmbAlgorithms.getSelectedIndex();
          suggestDestination();
       }//JComboBox??
    }
    public void itemStateChanged(ItemEvent e) {
       Object obj = e.getSource();
       if(obj instanceof JRadioButton){
          JRadioButton rbtn = (JRadioButton)obj;
          if(rbtn.isSelected()) btnAction.setText(rbtn.getText());
          suggestDestination();
       }// JRadioButton
    }//Item Listener
```

```
    void clearDetails(){
        txtStatus.setText(REFRESH);
        txtSource.setText("");
        txtDestination.setText("");
    }
    void suggestDestination(){
        if(txtSource.getText().length() == 0) return ;
        if(rbtnCompress.isSelected()){
            txtDestination.setText(txtSource.getText() +
extensionArray[algoSelected]);
        }else if(txtSource.getText().endsWith(extensionArray[algoSelected])){
            String buf =
txtSource.getText().substring(0,txtSource.getText().lastIndexOf(extensionArra
y[algoSelected]));
            txtDestination.setText(buf);
        }else {
            txtDestination.setText(txtSource.getText() +
extensionArray[algoSelected]);
        }
    }
}
```

//4: EveSplash.java

```
import java.awt.*;
import javax.swing.*;
import javax.swing.border.*;
public class EveSplash extends JWindow {
    private JLabel splashImage;
    private Dimension screensize;
    private JProgressBar progressBar;
    private int progress = 0;
    private int waitTime = 7;
    public EveSplash(JFrame parent){
        super(parent);
        splashImage=new JLabel(new
ImageIcon(getClass().getResource("/images/st.JPG")));
        progressBar = new JProgressBar(0,100);
        progressBar.setValue(0);
        getContentPane().add(splashImage, BorderLayout.CENTER);
        Border border = BorderFactory.createTitledBorder("Loading...");
        progressBar.setStringPainted(true);
        progressBar.setBorder(border);
        getContentPane().add(progressBar, BorderLayout.SOUTH);
        pack();
        screensize = Toolkit.getDefaultToolkit().getScreenSize();
```

```
        setLocation((screensize.width / 2) - (getSize().width /
2),(screensize.height / 2) - (getSize().height / 2));
        splashImage.setBorder(BorderFactory.createLineBorder(new Color(75,
75, 75)));
        setVisible(true);
        //doSplashStuff();
    }
    public void doSplashStuff(){
        final int pause = waitTime;
        final Runnable updateRunner = new Runnable() {
            public void run(){
                progressBar.setValue(progress);
            }
        };
        final Runnable closerRunner = new Runnable() {
            public void run() {
                setVisible(false);
            dispose();
            }
        };
        Runnable waitRunner = new Runnable() {
            public void run() {
                try {
                    int sec = pause;
```

```
                for(int i = 0;i<sec;i++){
                    progress += Math.round(Math.random()*(100/sec) + 1);
                    progress = Math.min(100,progress);
                    if(i >= sec - 1)
                        progress = 100;
                    SwingUtilities.invokeLater(updateRunner);
                    Thread.sleep(1000);
                }
                //Thread.sleep(1000); //one more for luck ;)
                SwingUtilities.invokeAndWait(closerRunner);
            }catch(Exception e){}
        }
    };
    Thread splashThread = new Thread(waitRunner, "SplashThread");
    splashThread.start();
  }
}
```

//5: EveFrame.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class EveFrame extends JFrame{
   private EveSplash splash;
```

```
    private JTabbedPane tabbedPane = new JTabbedPane();
    private EveComp  panCompression;
    private EveAbout panAbout;
    private JLabel  lblBanner;
    void centerWindow(){
        Dimension screensize = Toolkit.getDefaultToolkit().getScreenSize();
        setLocation((screensize.width / 2) - (getSize().width /
2),(screensize.height / 2) - (getSize().height / 2));
    }
    EveFrame(){
        setVisible(false);
        splash = new EveSplash(this);  //uncomment on release
        splash.doSplashStuff();
        setTitle("EVE - Compress Better, Compress Faster!");
        setSize(425,390);
        centerWindow();
        setLayout(new BorderLayout(5,5));
        lblBanner = new JLabel("SunZip",SwingConstants.CENTER);
        lblBanner.setSize(400,25);
        lblBanner.setFont(new Font("Monotype Corsiva",Font.BOLD &
Font.ITALIC,42));
        panCompression = new EveComp(this,false);
        panAbout = new EveAbout(this,false);
        tabbedPane.addTab("Compressor/Decompressor",panCompression);
```

```
        tabbedPane.addTab("About SunZip",panAbout);
        //The following line enables to use scrolling tabs.
tabbedPane.setTabLayoutPolicy(JTabbedPane.SCROLL_TAB_LAYOUT);
        getContentPane().add(lblBanner,BorderLayout.NORTH );
        getContentPane().add(tabbedPane,BorderLayout.CENTER);
        setResizable(false);
        setVisible(true);
    }
    @Override
    protected void processWindowEvent(WindowEvent e) {
        if (e.getID() == WindowEvent.WINDOW_CLOSING) {
            //System.exit(0); //remove on release
            int exit = JOptionPane.showConfirmDialog(this, "Are you
sure?","Confirm Exit?",JOptionPane.YES_NO_OPTION);
            if (exit == JOptionPane.YES_OPTION) {
                System.exit(0);
            }
        } else {
            super.processWindowEvent(e);
        }
    }
}
```

//6: EveGuiConstants.java

```
public interface EveGuiConstants {
    String[] algorithmNamesArray = {"Huffman Compression",
    //"Shannon Fano Compression",
    //"GZip Compression",
    //"Cosmo Compression",
    //"JunkCode Binary Compression",
    //"RLE Compressor",
    //"LZW Compressor"
    };
    String[] extensionArray = {".huf",
    //".sfe",
    ".gz",
    //".cos",
    //".jbe",
    //".rle",
    //".lzw"
    };
    final int COMP_HUFFMAN = 0;
    //final int COMP_SHANNONFANO = 1;
    //final int COMP_GZIP = 2;
    //final int COMP_COSMO = 3;
    //final int COMP_JBC = 4;
    //final int COMP_RLE = 5;
```

```
    //final int COMP_LZW = 6;
    final int COMPRESS = 32;
    final int DECOMPRESS = 33;
}
```

//7: EveWorkingDlg.java

```
import java.io.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
//compression algorithms
import CHuffmanCompressor.*;
public class EveWorkingDlg  extends JDialog implements
ActionListener,EveGuiConstants{
    private JFrame owner;
    private JProgressBar prgBar;
    private JButton btnCancel;
    private JLabel lblNote;
    private String gSummary = "";
    private String iFilename,oFilename;
    private boolean bCompress = false;
    private int algoSelected;
    void centerWindow(){
        Dimension screensize = Toolkit.getDefaultToolkit().getScreenSize();
```

```
        setLocation((screensize.width / 2) - (getSize().width /
2),(screensize.height / 2) - (getSize().height / 2));
    }
    EveWorkingDlg(JFrame parent){
        super(parent,true);
        owner = parent;
        setDefaultCloseOperation(JDialog.DO_NOTHING_ON_CLOSE);
        addWindowListener(new WindowAdapter() {
            @Override
            public void windowClosing(WindowEvent we) {
                //setTitle("Thwarted user attempt to close window.");
            }
        });
        setSize(300,120);
        centerWindow();
        buildDlg();
        setResizable(false);
        btnCancel.addActionListener(this);
        //setVisible(true);
    }
    void buildConstraints(GridBagConstraints gbc, int gx, int gy,int gw, int gh,
int wx, int wy) {
        gbc.fill = GridBagConstraints.HORIZONTAL;
        gbc.gridx = gx;
```

```
    gbc.gridy = gy;
    gbc.gridwidth = gw;
    gbc.gridheight = gh;
    gbc.weightx = wx;
    gbc.weighty = wy;
}
void buildDlg(){
    GridBagLayout gridbag = new GridBagLayout();
    GridBagConstraints constraints = new GridBagConstraints();
    constraints.anchor = GridBagConstraints.CENTER;
    setLayout(gridbag);
    prgBar = new JProgressBar();
    prgBar.setSize(100,30);
    prgBar.setStringPainted(false);
    prgBar.setIndeterminate(true);
    btnCancel = new JButton("Cancel");
    lblNote  = new JLabel("hahah",JLabel.CENTER);
    constraints.insets = new Insets(3,3,3,3);
    buildConstraints(constraints,1,0,2,1,50,30);
    gridbag.setConstraints(lblNote  ,constraints) ;
    add(lblNote);
    buildConstraints(constraints,0,1,4,1,100,40);
    gridbag.setConstraints(prgBar,constraints) ;
    add(prgBar);
```

```
        buildConstraints(constraints,1,2,2,1,50,30);
        constraints.fill = GridBagConstraints.NONE;
        gridbag.setConstraints(btnCancel ,constraints) ;
        add(btnCancel );
    }
    void doWork(String inputFilename,String outputFilename,int mode,int
algorithm){
        String buf;
        File infile = new File(inputFilename);
        //chk if file exists
        if(!infile.exists()){
            gSummary += "File Does not Exits!\n";
            return;
        }
        bCompress = (mode == COMPRESS);
        if(bCompress )
            lblNote.setText("Compressing " + infile.getName());
        else
            lblNote.setText("Decompressing " + infile.getName());
        setTitle(lblNote.getText());
        final int falgo = algorithm;
        iFilename = inputFilename;
        oFilename = outputFilename;
        gSummary = "";
```

```
        //Create Thread for Compress/Decompress
        final Runnable closeRunner = new Runnable(){
            public void run(){
                setVisible(false);
                dispose();
            }
        };
        Runnable workingThread = new Runnable(){
            public void run(){
                try{
                    boolean success = false;
                    switch(falgo){
                        case COMP_HUFFMAN :
                        if(bCompress){
                            CHuffmanEncoder he = new
CHuffmanEncoder(iFilename,oFilename);
                            success = he.encodeFile();
                            gSummary += he.getSummary();
                        }else{
                            CHuffmanDecoder hde = new
CHuffmanDecoder(iFilename,oFilename);
                            success = hde.decodeFile();
                            gSummary += hde.getSummary();
                        }
```

```
                    break;
                }
            }catch(Exception e){
                gSummary += e.getMessage();
            }
            try{
                SwingUtilities.invokeAndWait(closeRunner );
            }catch(Exception e){
                gSummary += "\n" + e.getMessage();
            }
        }
    };//working thread
    Thread work = new Thread(workingThread);
    work.start();
    setVisible(true);
  }
  public void actionPerformed(ActionEvent e) { //called only when cancel is
pressed
    //Object obj = e.getSource();
    dispose();
  }
  public String getSummary(){
    if(gSummary.length() > 0){
    String line = "-----------------------------------------";
```

```
        return line + "\n" + gSummary + line;
        }else return "";
    }
}
```

//8: CHuffmanDecoder.java

```
package CHuffmanCompressor;
import java.io.*;
import FileBitIO.*;
public class CHuffmanDecoder implements huffmanSignature{
    private String fileName,outputFilename;
    private String[] hCodes = new String[MAXCHARS];
    private int distinctChars = 0;
    private long fileLen=0,outputFilelen;
    private FileOutputStream outf;
    private String gSummary;
    public CHuffmanDecoder(){
        loadFile("","");
    }
    public CHuffmanDecoder(String txt){
        loadFile(txt);
    }
    public CHuffmanDecoder(String txt,String txt2){
        loadFile(txt,txt2);
```

```
}
public void loadFile(String txt){
    fileName = txt;
    outputFilename = stripExtension(txt,strExtension);
    gSummary = "";
}
public void loadFile(String txt,String txt2){
    fileName = txt;
    outputFilename = txt2;
    gSummary = "";
}
String stripExtension(String ff,String ext){
    ff = ff.toLowerCase();
    if(ff.endsWith(ext.toLowerCase())){
        return ff.substring(0,ff.length()-ext.length());
    }
    return "_" + ff;
}
public boolean decodeFile() throws Exception{
    if(fileName.length() == 0) return false;
    for(int i=0;i<MAXCHARS;i++) hCodes[i] = "";
    long i;
    CFileBitReader fin = new CFileBitReader(fileName);
    fileLen = fin.available();
```

```
String buf;
buf = fin.getBytes(hSignature.length());
if(!buf.equals(hSignature)) return false;
outputFilelen = Long.parseLong(fin.getBits(32),2);
distinctChars = Integer.parseInt(fin.getBits(8),2) + 1;
gSummary  += ("Compressed File Size : "+ fileLen + "\n");
gSummary  += ("Original   File Size : "+ outputFilelen + "\n");
gSummary  += ("Distinct   Chars     : "+ distinctChars + "\n");
for(i=0;i<distinctChars;i++){
    int ch = Integer.parseInt(fin.getBits(8),2);
    int len = Integer.parseInt(leftPadder(fin.getBits(5),8),2);
    hCodes[ch] = fin.getBits(len);
    //System.out.println((char)ch + " : "  + hCodes[ch]);
}
try{
    outf = new FileOutputStream(outputFilename);
    i = 0;
    int k;
    int ch;
    while(i < outputFilelen){
        buf = "";
        for(k=0;k<32;k++){
            buf += fin.getBit();
            ch  = findCodeword(buf);
```

```
                if(ch > -1){
                    outf.write((char)ch);
                    buf = "";
                    i++;
                    break;
                }
            }
            if(k >=32 )
                throw new Exception("Corrupted File!");
        }
        outf.close();
        return true;
    }catch(Exception e){
        throw e;
    }
}
int findCodeword(String cw){
    int ret = -1;
    for(int i=0;i<MAXCHARS;i++){
        if(hCodes[i] != "" && cw.equals(hCodes[i])){
            ret = i;
            break;
        }
    }
```

```
        return ret;
    }
    String leftPadder(String txt,int n){
        while(txt.length() < n )
            txt =  "0" + txt;
        return txt;
    }
    String rightPadder(String txt,int n){
        while(txt.length() < n )
            txt += "0";
        return txt;
    }
    public String getSummary(){
        return gSummary;
    }
}
```

//9: CHuffmanEncoder.java

```
package CHuffmanCompressor;
import java.io.*;
import FileBitIO.CFileBitWriter;
public class CHuffmanEncoder implements huffmanSignature{
    private String fileName,outputFilename;
    private HuffmanNode root = null;
```

```
private long[] freq  = new long[MAXCHARS];
private String[] hCodes = new String[MAXCHARS];
private int distinctChars = 0;
private long fileLen=0,outputFilelen;
private FileInputStream fin;
private BufferedInputStream in;
private String gSummary;
void resetFrequency(){
    for(int i=0;i<MAXCHARS;i++)
        freq[i] = 0;
    distinctChars = 0;
    fileLen=0;
    gSummary ="";
}
public CHuffmanEncoder(){
    loadFile("","");
}
public CHuffmanEncoder(String txt){
    loadFile(txt);
}
public CHuffmanEncoder(String txt,String txt2){
    loadFile(txt,txt2);
}
public void loadFile(String txt){
```

```
    fileName = txt;
    outputFilename = txt + strExtension;
    resetFrequency();
}
public void loadFile(String txt,String txt2){
    fileName = txt;
    outputFilename = txt2;
    resetFrequency();
}
public boolean encodeFile() throws Exception{
    if(fileName.length() == 0)
        return false;
    try{
        fin = new FileInputStream(fileName);
        in = new BufferedInputStream(fin);
    }catch(Exception e){ throw e; }
    //Frequency Analysis
    try {
        fileLen = in.available();
        if(fileLen == 0) throw new Exception("File is Empty!");
        gSummary += ("Original File Size : "+ fileLen + "\n");
        long i=0;
        in.mark((int)fileLen);
        distinctChars = 0;
```

```
    while (i < fileLen) {
        int ch = in.read();
        i++;
        if(freq[ch] == 0)
            distinctChars++;
        freq[ch]++;
    }
    in.reset();
}
catch(IOException e) {
    throw e;
    //return false;
}
gSummary += ("Distinct Chars " + distinctChars + "\n");
/*
System.out.println("distinct Chars " + distinctChars);
//debug
for(int i=0;i<MAXCHARS;i++){
    if(freq[i] > 0)
        System.out.println(i + ")" + (char)i + " : " + freq[i]);
}
*/
CPriorityQueue  cpq = new CPriorityQueue (distinctChars+1);
int count  = 0 ;
```

```
        for(int i=0;i<MAXCHARS;i++){
            if(freq[i] > 0){
                count ++;
                //System.out.println("ch = " + (char)i + " : freq = " + freq[i]);
                HuffmanNode np = new HuffmanNode(freq[i],(char)i,null,null);
                cpq.Enqueue(np);
            }
        }
        //cpq.displayQ();
        HuffmanNode low1,low2;
        while(cpq.totalNodes() > 1){
            low1 = cpq.Dequeue();
            low2 = cpq.Dequeue();
            if(low1 == null || low2 == null) { throw new Exception("PQueue Error!");
}
            HuffmanNode intermediate = new
HuffmanNode((low1.freq+low2.freq),(char)0,low1,low2);
            if(intermediate == null) { throw new Exception("Not Enough Memory!");
}
            cpq.Enqueue(intermediate);
        }
        //cpq.displayQ();
        //root = new HuffmanNode();
        root = cpq.Dequeue();
```

```
        buildHuffmanCodes(root,"");
        for(int i=0;i<MAXCHARS;i++) hCodes[i] = "";
        getHuffmanCodes(root);
        /*
        //debug
        for(int i=0;i<MAXCHARS;i++){
            if(hCodes[i] != ""){
                System.out.println(i + " : " + hCodes[i]);
            }
        }
        */
        CFileBitWriter hFile = new CFileBitWriter(outputFilename);
        hFile.putString(hSignature);
        String buf;
        buf = leftPadder(Long.toString(fileLen,2),32); //fileLen
        hFile.putBits(buf);
        buf = leftPadder(Integer.toString(distinctChars-1,2),8); //No of Encoded
Chars
        hFile.putBits(buf);
        for(int i=0;i<MAXCHARS;i++){
            if(hCodes[i].length() != 0){
                buf = leftPadder(Integer.toString(i,2),8);
                hFile.putBits(buf);
                buf = leftPadder(Integer.toString(hCodes[i].length(),2),5);
```

```
            hFile.putBits(buf);
            hFile.putBits(hCodes[i]);
        }
    }
    long lcount = 0;
    while(lcount < fileLen){
        int ch = in.read();
        hFile.putBits(hCodes[(int)ch]);
        lcount++;
    }
    hFile.closeFile();
    outputFilelen =  new File(outputFilename).length();
    float cratio = (float)(((outputFilelen)*100)/(float)fileLen);
    gSummary += ("Compressed File Size : " + outputFilelen + "\n");
    gSummary += ("Compression Ratio : " + cratio + "%" + "\n");
    return true;
}
void buildHuffmanCodes(HuffmanNode parentNode,String parentCode){
    parentNode.huffCode = parentCode;
    if(parentNode.left != null)
    buildHuffmanCodes(parentNode.left,parentCode + "0");
    if(parentNode.right != null)
    buildHuffmanCodes(parentNode.right,parentCode + "1");
}
```

```
void getHuffmanCodes(HuffmanNode parentNode){
    if(parentNode == null)
        return;
    int asciiCode = (int)parentNode.ch;
    if(parentNode.left == null || parentNode.right == null)
        hCodes[asciiCode] = parentNode.huffCode;
    if(parentNode.left != null )
        getHuffmanCodes(parentNode.left);
    if(parentNode.right != null )
        getHuffmanCodes(parentNode.right);
}
String leftPadder(String txt,int n){
    while(txt.length() < n )
        txt =  "0" + txt;
    return txt;
}
String rightPadder(String txt,int n){
    while(txt.length() < n )
        txt += "0";
    return txt;
}
public String getSummary(){
    return gSummary;
}
```

```
}
```

//10: CPriorityQueue.java

```
package CHuffmanCompressor;
class CPriorityQueue{
    final int DEFAULTMAX = 256;
    HuffmanNode [] nodes;
    int capacity,total;
    public CPriorityQueue(){
        capacity = DEFAULTMAX;
        total = 0;
        nodes = new HuffmanNode[capacity];
    }
    public CPriorityQueue(int max){
        capacity = max;
        total = 0;
        nodes = new HuffmanNode[capacity];
    }
    public boolean Enqueue(HuffmanNode np){
        if(isFull())
            return false;
        if(isEmpty()){ //first element?
            nodes[total++] = np;
            return true;
```

```
    }
    int i = total-1,pos;
    while(i >= 0){
        if(nodes[i].freq < np.freq){
            break;
        }
        i--;
    }
    pos = total-1;
    while(pos >= i+1){
        nodes[pos+1] = nodes[pos];
        pos--;
    }
    nodes[i+1] = np;
    total++;
    return true;
}
public HuffmanNode Dequeue(){
    if(isEmpty())
        return null;
    HuffmanNode ret = nodes[0];
    total--;
    for(int i = 0;i<total;i++)
        nodes[i] = nodes[i+1];
```

```
        return ret;
    }
    public boolean isEmpty(){
        return (total == 0);
    }
    public boolean isFull(){
        return (total == capacity);
    }
    public int totalNodes(){
        return total;
    }
    //debug
    public void displayQ(){
        for(int i=0;i<total;i++){
            System.out.println("Q" + i + ") " + nodes[i].ch + " : " + nodes[i].freq);
        }
    }
}
```

//11: HuffmanNode.java

```
package CHuffmanCompressor;
public class HuffmanNode{
    HuffmanNode left,right;
    public long freq;
```

```
    public char ch;
    //Code Words
    public String huffCode;
    public HuffmanNode(){
        freq = 0;
        ch = 0;
        huffCode = "";
        left = null;
        right = null;
    }
    public HuffmanNode(long xfreq,char xch,HuffmanNode lchild,HuffmanNode
rchild){
        freq = xfreq;
        ch = xch;
        left = lchild;
        right = rchild;
        huffCode = "";
    }
}
```

//12: huffmanSignature.java

```
package CHuffmanCompressor;
public interface huffmanSignature{
    final String hSignature = "SHE";
```

```
    final int MAXCHARS = 256;
    final String strExtension = ".huf";
}
```

//13: CFileBitReader.java

```
package FileBitIO;
import java.io.*;
//File Bit Reader - Read Files BitWise
public class CFileBitReader {
    private String fileName;
    private File inputFile;
    private FileInputStream fin;
    private BufferedInputStream in;
    private String currentByte;
    public CFileBitReader() throws Exception{
        try{
            fileName = "";
            //loadFile(fileName);
        }catch(Exception e){
            throw e;
        }
    }
    public CFileBitReader(String txt) throws Exception{
        try{
```

```
            fileName = txt;
            loadFile(fileName);
        }catch(Exception e){
            throw e;
        }
    }
    public boolean loadFile(String txt) throws Exception{
        fileName = txt;
        try {
            inputFile = new File(fileName);
            fin = new FileInputStream(inputFile);
            in = new BufferedInputStream(fin);
            currentByte = "";
            return true;
        }catch(Exception e){
            throw e;
        }
        //return true;
    }
    String leftPad8(String txt){
        while(txt.length() < 8 )
            txt =  "0" + txt;
        return txt;
    }
```

```
String rightPad8(String txt){
    while(txt.length() < 8 )
        txt +=  "0";
    return txt;
}
public String getBit() throws Exception{
    try{
        if(currentByte.length() == 0 && in.available() >= 1){
            int k = in.read();
            currentByte = Integer.toString(k,2);
            currentByte = leftPad8(currentByte);
        }
        if(currentByte.length() > 0){
            String ret = currentByte.substring(0,1);
            currentByte = currentByte.substring(1);
            return ret;
        }
        return "";
    }catch(Exception e){throw e;}
}
public String getBits(int n) throws Exception{
    try{
        String ret = "";
        for(int i=0;i<n;i++){
```

```
            ret += getBit();
        }
        return ret;
    }catch(Exception e){throw e;}
}
public String getBytes(int n) throws Exception{
    try{
        String ret = "",temp;
        for(int i=0;i<n;i++){
            temp = getBits(8);
            char k = (char)Integer.parseInt(temp,2);
            ret += k;
        }
        return ret;
    }catch(Exception e){throw e;}
}
public boolean eof() throws Exception{
    try{
        return (in.available()== 0);
    }catch(Exception e){throw e;}
}
public long available() throws Exception{
    try{
        return in.available();
```

```
        }catch(Exception e){throw e;}
    }
    public void closeFile() throws Exception{
        try{
            in.close();
        }catch(Exception e){throw e;}
    }
}
```

//14: CFileBitWriter.java

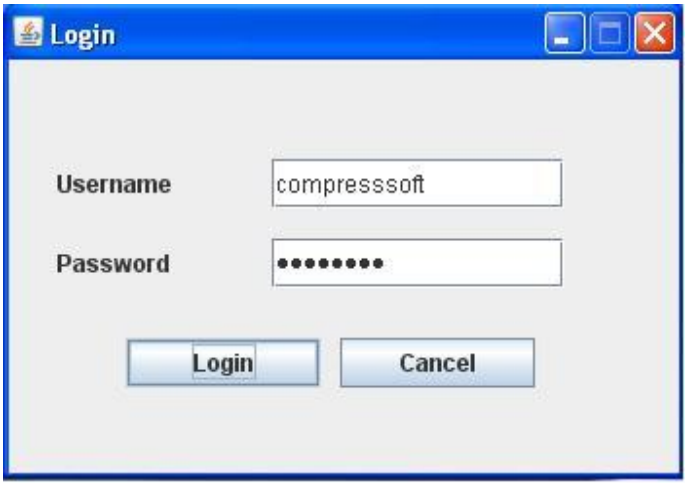
```
package FileBitIO;
import java.io.*;
//File Bit Writer - Write to Files BitWise
public class CFileBitWriter {
    private String fileName;
    private File outputFile;
    private FileOutputStream fout;
    private BufferedOutputStream outf;
    private String currentByte;
    public CFileBitWriter() throws Exception{
        try{
            fileName = "";
            //loadFile(fileName);
        }catch(Exception e){
```

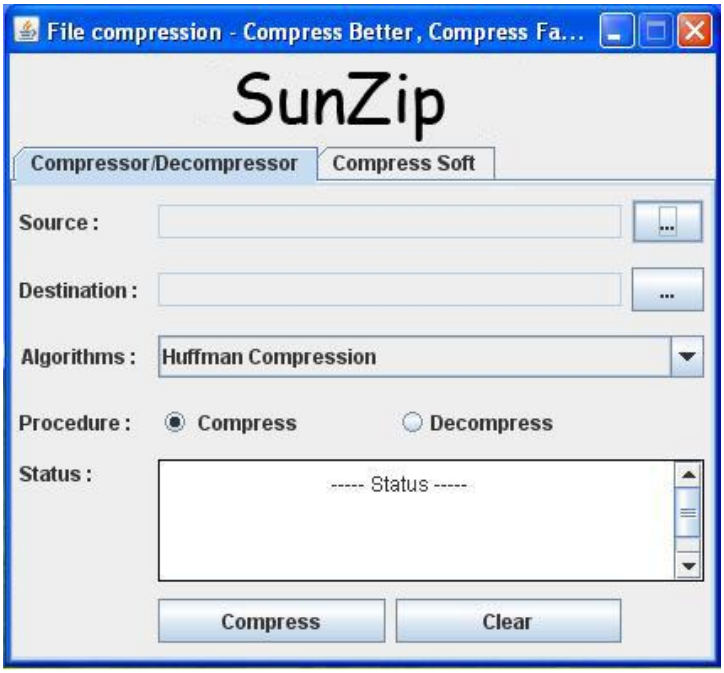
```
            throw e;
        }
    }
    public CFileBitWriter(String txt) throws Exception{
        try{
            fileName = txt;
            loadFile(fileName);
        }catch(Exception e){
            throw e;
        }
    }
    public boolean loadFile(String txt) throws Exception{
        fileName = txt;
        try {
            outputFile = new File(fileName);
            fout = new FileOutputStream(outputFile);
            outf = new BufferedOutputStream(fout);
            currentByte = "";
            return true;
        }catch(Exception e){
            throw e;
        }
        //return true;
    }
```

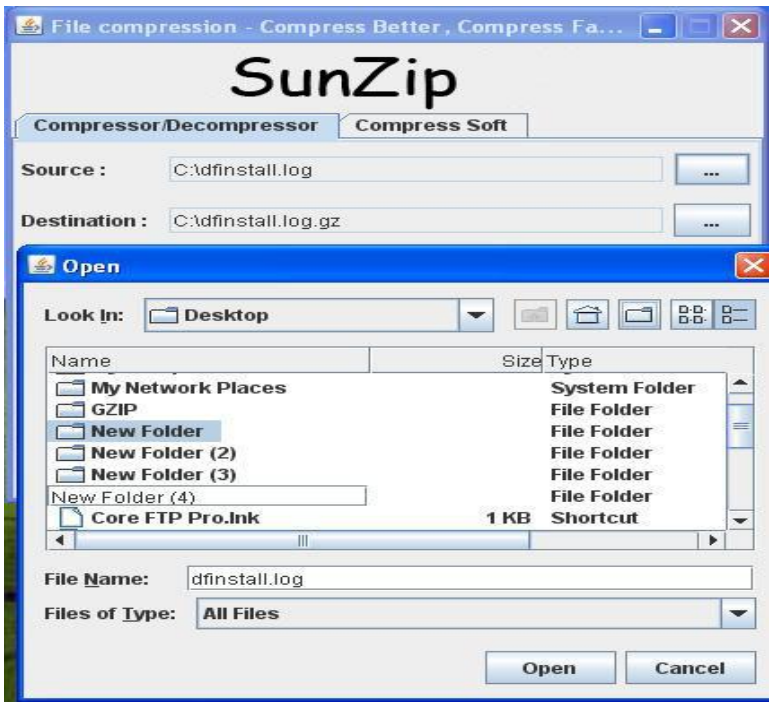
```
public void putBit(int bit) throws Exception{
    try{
        bit = bit % 2;
        currentByte = currentByte + Integer.toString(bit);
        if(currentByte.length() >= 8){
           int byteval = Integer.parseInt(currentByte.substring(0,8),2);
            outf.write(byteval);
            currentByte = ""; //reset
        }
    }catch(Exception e){throw e;}
}
public void putBits(String bits) throws Exception{
    try{
        while(bits.length() > 0){
            int bit = Integer.parseInt(bits.substring(0,1));
            putBit(bit);
            bits = bits.substring(1);
        }
    }catch(Exception e){throw e;}
}
public void putString(String txt) throws Exception{
    try{
        while(txt.length() > 0){
            String binstring = Integer.toString(txt.charAt(0),2);
```

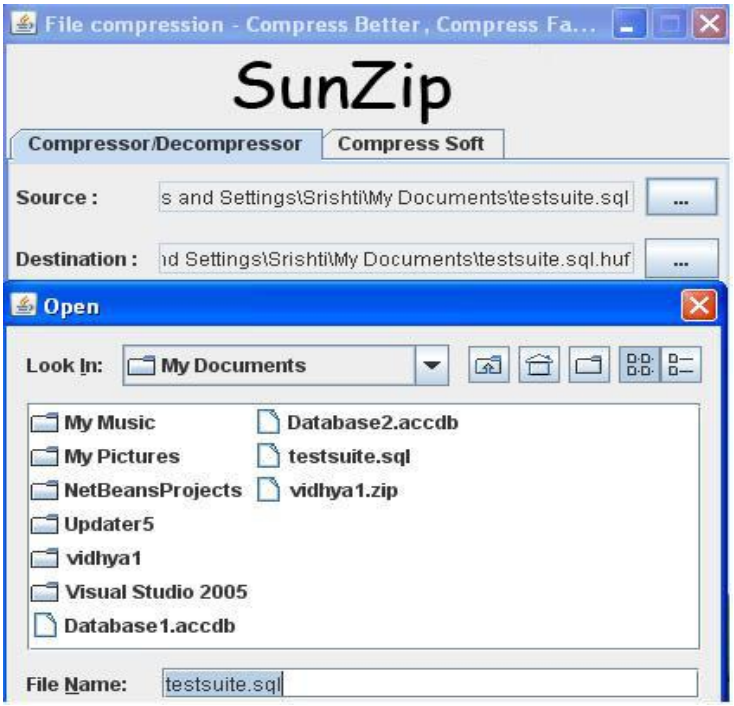
```
            binstring = leftPad8(binstring );
            putBits(binstring);
            txt = txt.substring(1);
        }
    }catch(Exception e){throw e;}
}
String leftPad8(String txt){
    while(txt.length() < 8 )
        txt =  "0" + txt;
    return txt;
}
String rightPad8(String txt){
    while(txt.length() < 8 )
        txt +=  "0";
    return txt;
}
public void closeFile() throws Exception{
    try{
        //check if incomplete byte exists
        while(currentByte.length() > 0){
            putBit(1);
        }
    outf.close();
    }catch(Exception e){ throw e;}
```

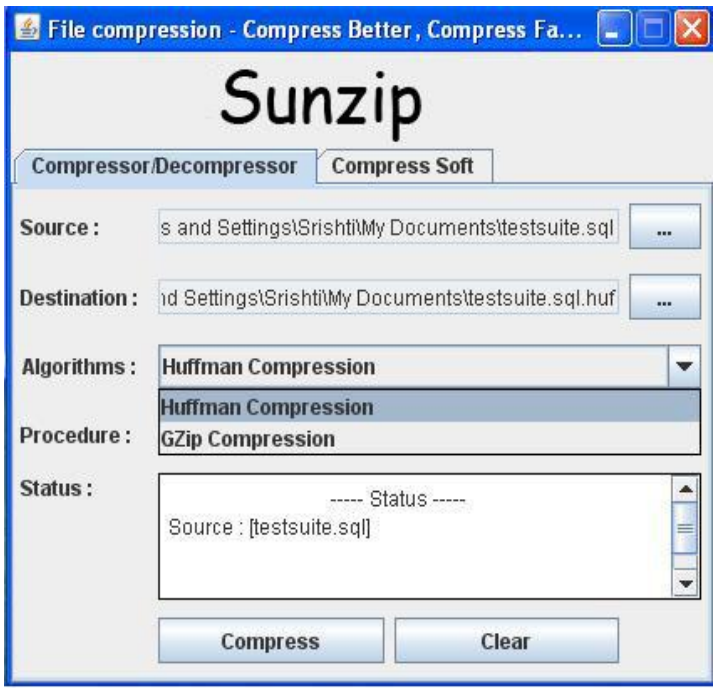
SCREENS AND RESULTS

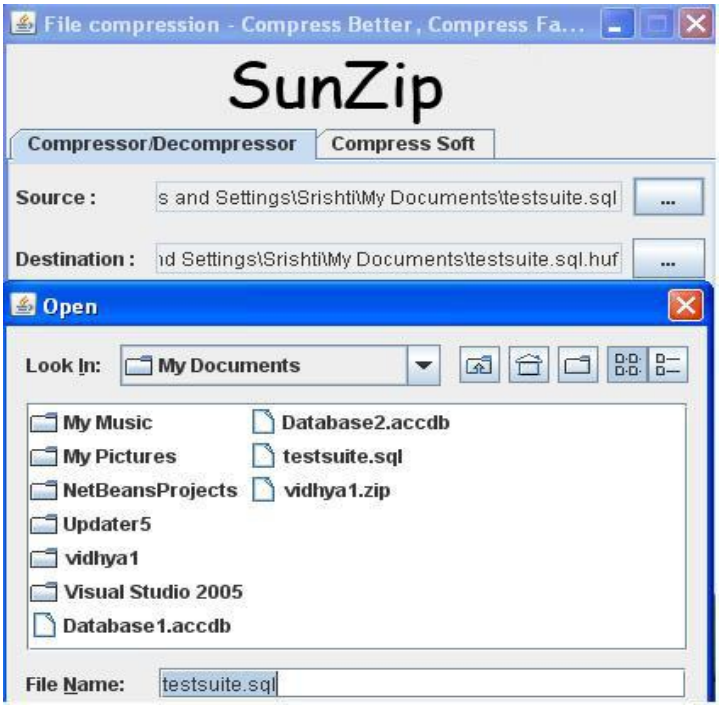


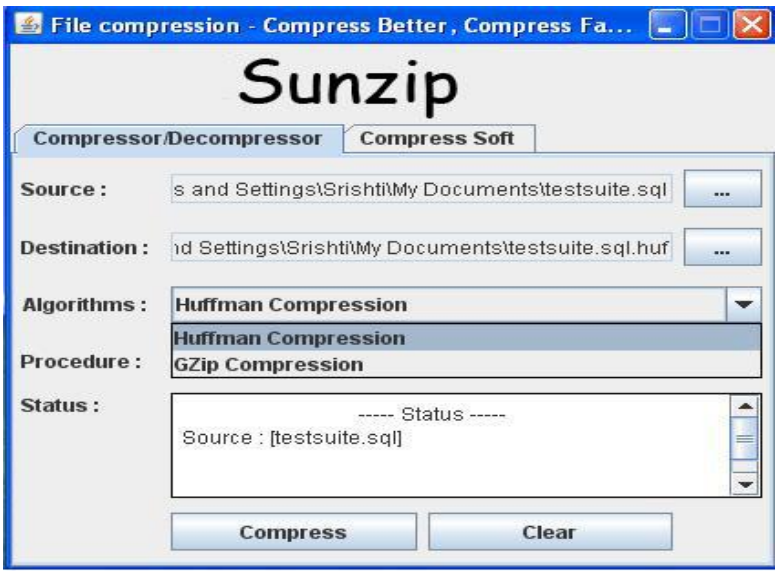


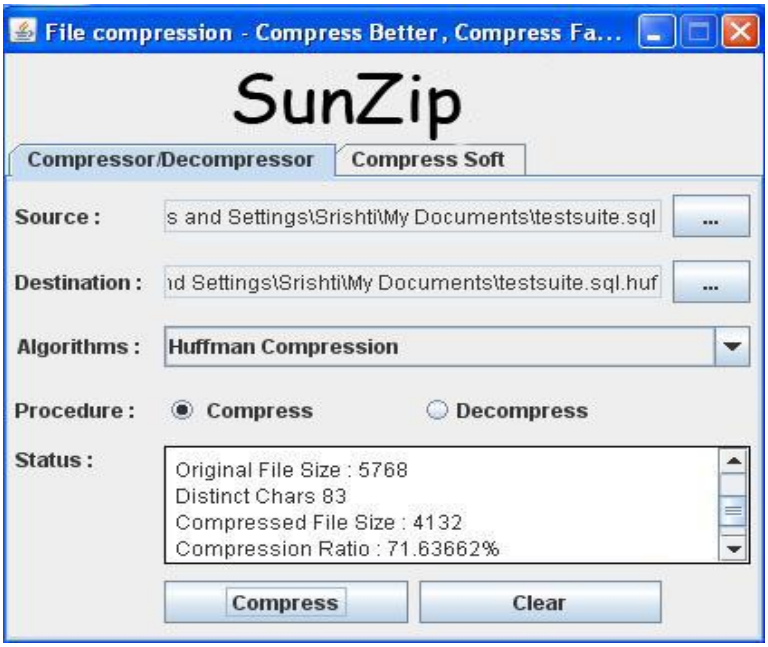


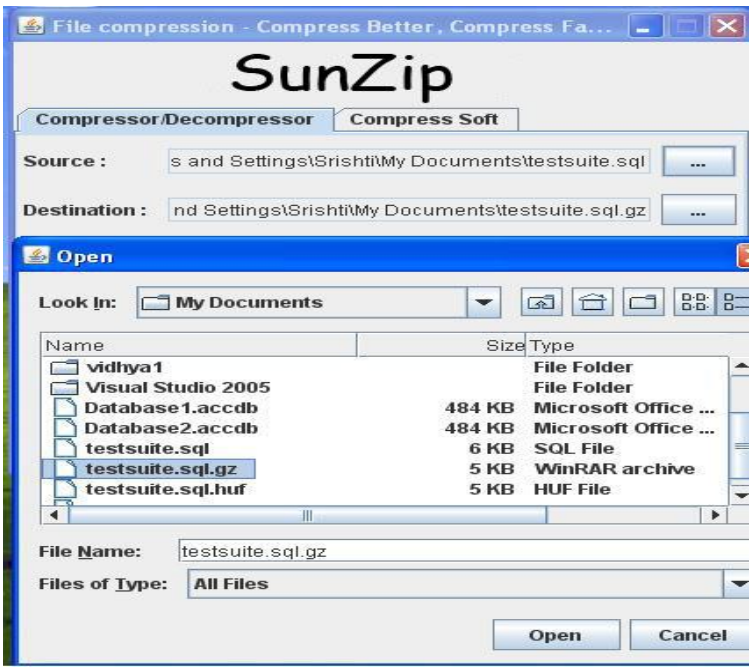


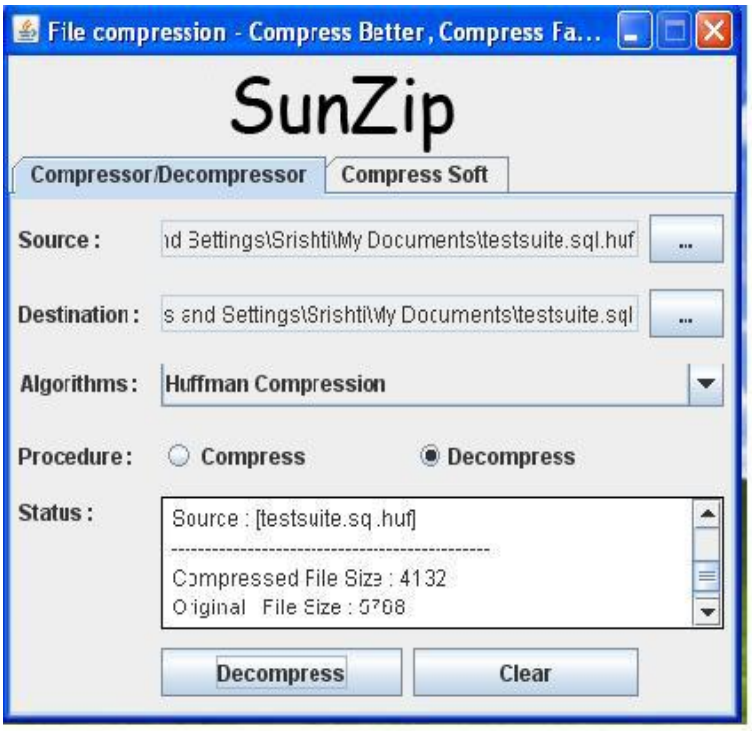












Chapter-10

System Testing

SYSTEM TESTING

10.1. INTRODUCTION

Testing is a process, which reveals errors in the program, it is the major quality measure employed during software development. During testing , the program is executed with a set of test cases and the output of the program for the test cases is evaluated to determine if the program is performing as it is expected to perform.

10.2. TESTING PROCESS

Testing is a process of executing a program with the interest of finding an error. A good test is one that has high probability of finding the yet undiscovered error. Testing should systematically uncover different classes of errors in a minimum amount of time with a minimum amount of efforts.

Two classes of inputs are provided to test the process

1. A software configuration that includes a software requirement specification, a design specification and source code.
2. A software configuration that includes a test plan and procedure, any testing tool and test cases and their expected results.

10.3. LEVELS OF TESTING

In order to make sure that the system does not have errors, the different levels of testing strategies that are applied at different phases of software development. They are:

1. UNIT TESTING

Unit test comprises of a set tests performed by an individual program prior to the integration of the unit into large system. A program unit is usually the smallest free functioning part of the whole system. Module unit testing should be as exhaustive as possible to ensure that each representation handled by each module has been tested. All the units that makeup the system must be tested independently to ensure that they work as required.

During unit testing some errors were raised and all of them were rectified and handled well. The result was quiet satisfactory and it worked well.

2. INTEGRATION TESTING

Integration testing is a system technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. The objective is to take unit tested modules and build a program structure that has been dictated by design. Bottom-up integration is the traditional strategy used to integrate the components of a software system into functioning whole. Bottom-up integration consists of unit test followed by testing of the entire system. A sub-system consists of several modules that communicated with other defined interface.

The system was done the integration testing. All the modules were tested for their compatibility with other modules .They test was almost successful. All the modules coexisted very well, with almost no bugs. All the modules were encapsulated very well so as to not hamper the execution of other modules.

3. VALIDATION TESTING

After validation testing, software is completely assembled as a package, interfacing errors that have been uncovered and corrected and the final series of software test; the validation test begins. Steps taken during software design and testing can greatly improve the probability of successful integration in the larger system. System testing is actually a series of different tests whose primary purpose is to fully exercise the compute –based system.

4. RECOVERY TESTING

It is a system that forces the software to fail in a variety of ways and verifies that the recovery is properly performed.

5. SECURITY TESTING

It attempts to verify that protection mechanisms built into a system will in fact protect it from improper penetration. The system's security must of course be tested from in vulnerability form frontal attack.

6. STRESS TESTING

Stress tools are designed to confront programs with abnormal situations. Stress testing executes a system in a manner that demands resources in abnormal quantity and volume.

7. BLACK BOX TESTING

Black box testing is done to find out the following information as shown in below:

- Incorrect or missing functions
- Interface errors

- Errors or database access
- Performance error
- Termination error

The mentioned testing is carried out successfully for this application according to the user's requirement specification.

8. TEST DATA OUTPUT

After preparing test data, the system under study is tested using the test data. While testing the system using test data, errors are again uncovered and corrected by using above testing and corrections are also noted for future use.

Chapter-11

Conclusions and Future Enhancements

CONCLUSIONS AND FUTURE ENHANCEMENTS

The project SunZip is completed, satisfying the required design specifications. The system provides a user-friendly interface. The software is developed with modular approach. All modules in the system have been tested with valid data and invalid data and everything work successfully. Thus the system has fulfilled all the objectives identified and is able to replace the existing system. The constraints are met and overcome successfully. The system is designed as like it was decided in the design phase. The system is very user friendly and will reduce time consumption. This software has a user-friendly screen that enables the user to use without any inconvenience. The user need not depend on third party software's like winzip, winrar, Stuff etc. The software can be used to compress files and they can be decompressed when the need arises. The application has been tested with live data and has provided a successful result. Hence the software has proved to work efficiently.

Chapter-12

Bibliography / References

BIBLIOGRAPHY / REFERENCES

- Charles Hampfed : (2000) 'Basic JAVA' University of Toronto
- Herhert Schildt : The Complete Reference Java2 (Fifth Edition)
- E. Balaguruswamy : Core Java
- Jim Farely $ William Crawford : Java Enterprise (Third edition)