Software Requirements Specification

Version 1.0

Group Id: **F10204017**

Supervisor Name :< Tanweer Arshad**>**

Revision History

Date (dd/mm/yyyy)	Version	Description	Author
25-11-2010	1.0	Introduction of the project	Write student(s) id mc080402021

Table of Contents

SRS Document

Scope of Project:

Project is divided into two parts

1.Email message security

It contains the following functionalities.
It will take the text in the form of the message.
It will encrypt to the give user data in the form of the encrypted form by the setting some key that is private to users who has created the message for someone other.

It will then set the given message on the user selected uploaded picture and then set the encrypted message and the private key upon the message picture and automatically pass phrase is set on the picture and the user having only the particular email address can only read that particular.

2.Security of File on the Hard

This will provide the following facilities having the multiple users on the single PC. The users are logged on the computer if they are more then on its need to secure the text files and the folders of the users from the others so that one can feel safe and ease at home so, It provides the functionality of converting of the some particular documents or text file to another image having the secure pass phrase.

It provides the facility of the user to secure his folder and break to other one from reaching that folder and don’t let know about he secret are laid in the form of the files and directory in the particular Folder

Data Encryption Compression Simulator is started to develop by keeping the thing in the mind that to provide user different kinds of functionalities related to the transformation through the internet in the secure manner so that user can share and transfer any information to the other user of the world in different ways such as by file transferring, by chatting, by using discussion form etc, but due to time lamination, all these are not in the specification of the project.

The project specification includes only he E-mail messages security and any other person can not read your information. So in the future intended the to add other functionalities or the chatting and discussion form but now indented to develop the above mentioned two functionality so that in the different style to the others which are already there to do this job.

For example, if complying with a standard such as the Payment Card Industry's (PCI) Data Security Standard, you must carefully read and understand its requirements. The PCI Data Security Standard has detailed encryption requirements as well as requirements for managing the encryption key. If your implementation does not follow these requirements, it will be out of compliance and you may fail your PCI audit.

Many of you are considering encryption—especially of backup tapes—to avoid the state notification law requirement of notifying individuals when their personal data is lost or stolen. Even if the law or regulation doesn't specify detailed encryption requirements (such as which encryption algorithm must be used to avoid notification requirements), I encourage you to architect your implementation as though you were meeting a standard such as the PCI's. You might as well implement a strong encryption scheme with strong

key management processes in case the laws change and start specifying strict requirements (or your business changes and you suddenly find yourself having to comply with a strict law or regulation). In other words, if you're going to the trouble of implementing encryption, you might as well do it right.

For example, if your organization stores credit card numbers, the PCI Data Security Standard dictates that the transmission of credit card information must be encrypted as well as any credit card information that is "at rest" (stored.) If you're storing protected electronic healthcare information, HIPAA dictates what needs to be encrypted. It recommends that data at rest be encrypted but doesn't require it. It does, however, require that data transmitted outside of the organization's network be encrypted. Your organization might have a trading partner with encryption standards required to fulfill its service agreement. Finally, the data owners may specify that all backups of their data need to be encrypted because of the privacy of the data (for example, social security numbers or bank account numbers) or its importance to the company (intellectual property, specialty vendors, etc.).

Determining what data to encrypt is the easy step. When specific data residing in a database is to be encrypted, the more difficult task is determining all—and I do mean *all*—of the places the data resides. Without fail, private or confidential data resides in more places than first come to mind. To ensure the location list is complete, you need to assemble selected programmers, system administrators, data owners (such as the lead HR administrator), database administrators, system analysts, and perhaps representatives from your third-party software vendors. Beyond finding all of the physical files in which the data resides, you can't forget the private information residing in spooled files, outfiles, query files, text files, stream files, and printed reports. Also make note of databases replicated to high availability (HA) machines, databases residing on development machines (including copies of the production database made into developers' libraries), logical files that include the fields needing to be encrypted, and data propagated to network servers or data warehouses along with desktop applications such as Excel.

After listing all of the places where the private data resides, examine each instance and

determine whether the private data is actually required for that specific instance or can be eliminated. In many cases, you can significantly reduce the scope of your project (as well as the exposure to your organization's private data) if you stop storing the private data on servers, stop allowing it to be included in queries, create a logical file that does not include the fields with private data, and eliminate the private data from printed reports and Excel spreadsheets. Often, the data is included in these places simply because it's part of the information in the original file structure, not because it's actually necessary.

Here are a couple of examples to consider. Do credit card numbers really need to be included on a printed report of all returns to a store if the person viewing the report is interested only in what's being returned and the reason for the return? Doubtful. Without question, removing the credit card number from the report is much easier than adding logic to the application to allow this individual to first encrypt this instance of the data and then decrypt it when the report is generated. It also eliminates the problems of putting appropriate access controls on the file, adding routines to encrypt the credit card number, and dealing with the proper disposal (shredding) of the printed reports.

Should the entire employee master file be available to all employees? No, but it often is because the employee extensions are in the file and the online phone directory needs access to this information. To solve this issue, the solution is to either move the phone extensions to another file or provide a logical file over the employee master that provides only the employee extensions, not all fields in the file.

For example, if the data is a social insurance number (in Canada) or a driver's license number used only for verification purposes, you can likely store it in a one-way hash. A hash cannot return the original data; it's one-way only. Using a hash algorithm is more simple and more secure than implementing more-complex encryption algorithms that allow decryption of data.

To ensure encryption provides an additional layer of protection on top of the access controls you have on database files, you must limit the scenarios where the data is

decrypted. For example, if the data is being sent to a bank for transaction processing or to a clearinghouse for verification, clear text data is required. For this scenario, determine how and when to decrypt the data so that the time the data is in clear text is as short as possible. Also, once the clear text data has been transmitted over an encrypted session, the clear text data needs to be removed or cleared as soon as possible.

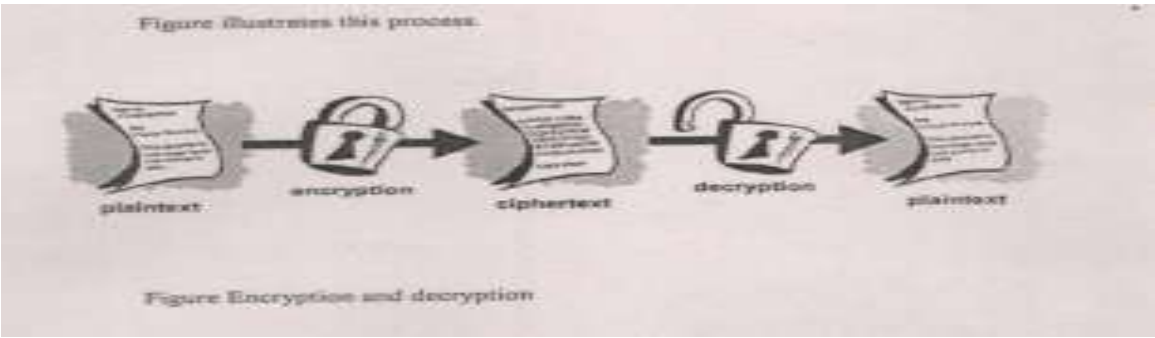
Functional and non Functional Requirements:

<Write here in detail about the functional and non-functional requirements of your system in separate headings>

FUNCTIONAL REQUIREMNTS

ENCRYPTION AND DECRYPTION

Data that ca be read and understood without any special measures is called plain text. The method of disguising plaintext is such a way as to hide its substance is called encryption. Encryption plaintext results in unreadable gibberish called cipher text. You are encryption to ensure that information is hidden from anyone for whom it is not intended, even those who can see the encrypted data. The process of reverting cipher text to its original plaintext is call decryption.



What is cryptography?

Cryptography is the science of using mathematics to encrypt and decrypt data. Cryptography enables you to store sensitive information or transmit it across in secure network (like the internet) so that it cannot be ready by any one except the intended recipient.

While cryptography is the science of securing data, cryptanalysis is the science of analyzing and breaking secure communication.Classical cryptanalysis involves and breaking secure combination. Classical cryptanalysis involves and interesting combination of analytical reasoning, application of mathematical tools, pattern finding, patience, determination and luck. cryptanalysis are also called attackers. cryptology embraces the cryptography and cryptanalysis

STRONG CRYPTOGRPHY.

There are two kind of cryptography in this world cryptography that will stop your kid from reading your files and cryptography that will stop major governments from reading your files. This book is about the letter.

Cryptography can be strong or weak as explained above. Cryptographic strength is measure in the time and resources it would require to recover the plaintext. The result of strong cryptography is cipher text that is very difficult to decipher without possession of the appropriate decoding tool.
How difficult?

Given all today's computing power and available time even a billion computers doing a billion checks a second it not possible to decipher the result of strong cryptography

before the end of the universe. One would think, then that strong cryptography would hold up rather well against even an extremely determined cryptanalysis. Who's really to say? no one has proven that the strongest encryption obtainable today will told up under tomorrow's computing power.

However, the strong cryptography employed by

Data Encryption Compression Simulator is the best available today.
Vigilance and conservation will protect you better, however, than claims of impenetrability.

How does cryptography work?

A cryptographic algorithm or cipher is mathematical function used in the encryption and decryption process. A cryptographic algorithm works in combination with a key word, number, or phrase to encrypt the plaintext. The same plaintext encrypts to different cipher text with different keys. The security of encrypted data is entirely dependent on two things. The strength of the cryptographic algorithm and secrecy of the key. A cryptographic algorithm, pulls all possible keys and all the protocols that make it work comprise a cryptosystem. Data Encryption Simulator is a cryptosystem.

Conventional Cryptography

In conventional cryptography, also called secret key or symmetric key encryption, one key is used both for encryption and decryption. The data Encryption Standard (DES) is an example of conventional cryptosystem that is widely employed by the Federal Government.

An illustration of the conventional encryption process is

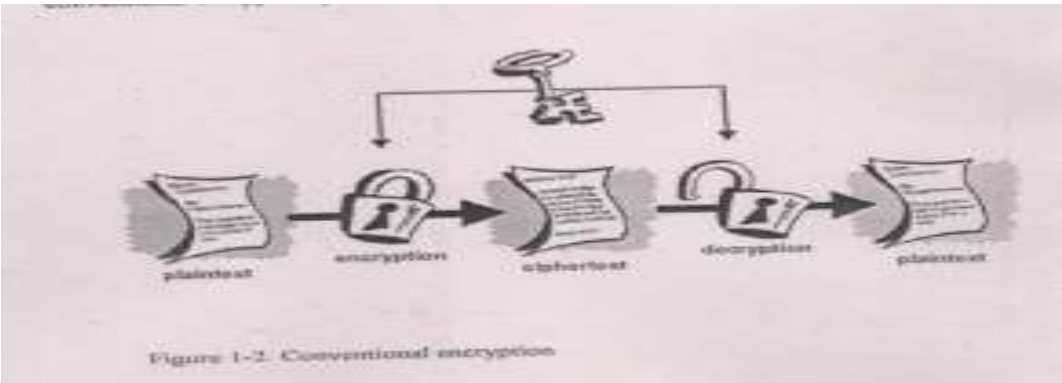


Figure 1-2. Conventional encryption

Caesar's Cipher

An extremely simple example of conventional cryptography is a substitution cipher. A substitution cipher substitutes one piece for another. This is most frequently done by offsetting letters of the alphabet.

Two examples are Captain Midnight's Secret Decoder Ring, which you may have owned when you were a kid, and Julius Caesar's cipher. In both cases, the algorithm is to offset the alphabet and the key is the number of characters to offset it.

For example, if we encode the word "SECRET" using Caesar's kay value of 3, we offset the alphabet so that the 3rd letter down (D) begins the alphabet. So starting

ABCDEFGHIJKLMNOPQRSTUVWXYZ

And sliding everything up by 3,you get

DEFHUKOLOMONPIQUCYIOPRQUCA

Where D=A ,E=B,F=C and so on.

Using this scheme, the plaintext “SECRET” encrypts as “VHFUHW”. To allow someone else to read the cipher text, you tell them that the key is 3.Obviously,this exceedingly weak cryptography by today’s standards, but they ,it worked for Caser, and it also illustrates how conventional cryptography works.

Key management and conventional encryption

Conventional encryption has benefits. It is very fast. It is especially useful for encrypting data that is not going anywhere. However, conventional encryption alone as a means fro transmitting secure data can be quite expensive simply due to the difficulty of secure key distribution. Recall a character from your favorite spy movie: the person with a locked briefcase handcuffed to his or her wrist. What is in the briefcase? Anyway?.Its probably not missile launch code/biotoxin formula/invasion plan itself. It’s the key that will decrypt the secret data.

For sender and recipient to communicate securely using conventional encryption. they must agree upon a key and keep it secret between themselves. If they are in different physical locations,they must trust a courier, the Bat phone or some other secure communication medium to prevent the disclosure of the secret key during transmission. Anyone who overhears or intercepts the key in transit can later read, modify and forge all information encrypted or authenticated with that key. From DES Captain Midnight’ secret Decoder Ring, the persistent problem with conventional encryption is key distribution. How do you et the key to recipient without some intercepting.

PUBLIC KEY CRYPTOGRAPHY

The problem of key distribution are solved by the key cryptography, the concept of which was introduced by Whitfield Diffie and Martin Hellman in 1975.Public Key cryptography is an asymmetric scheme that uses a pair of keys for encryption a public key, which encrypts data and a corresponding private or secret key for description. You publish your public key can then encrypt information that only you can read. Even people you have nave met.

It is computationally infeasible to deduce the private key from the public key.Anyone who has a public key can encrypt information but cannot decrypt it. Only the person who has the corresponding private key can decrypt the information.

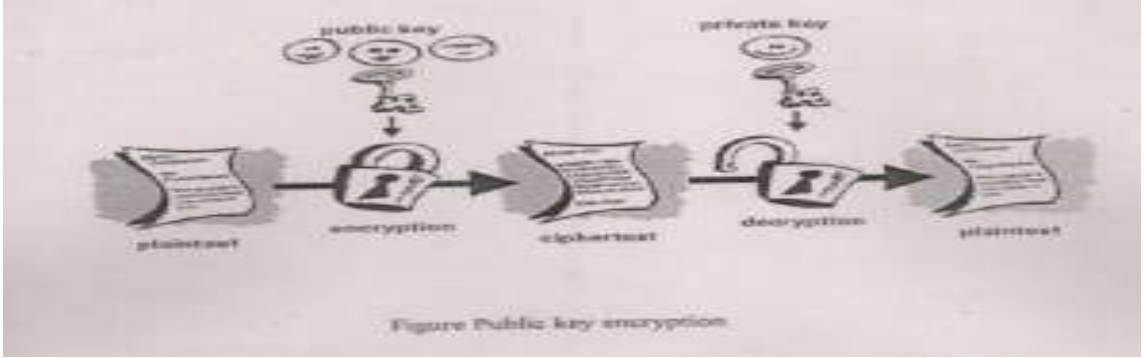


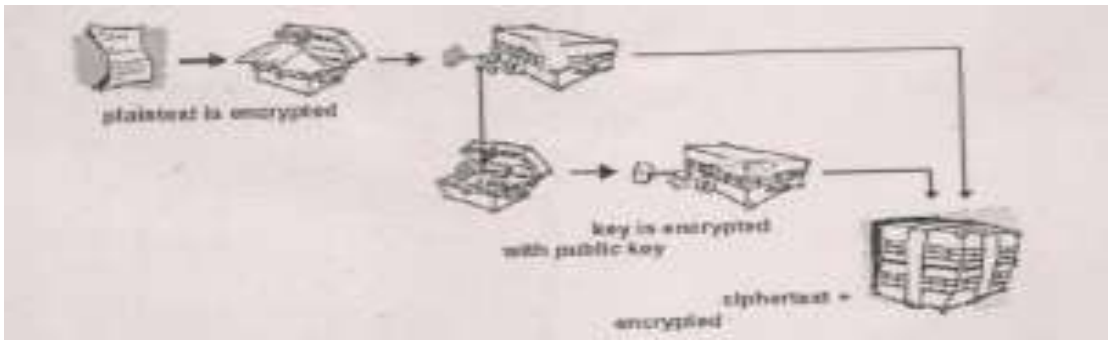
Figure Public key encryption

The primary benefit of public key cryptography is that it allows people who have no preexisting security arrangements to exchange messages securely. The need for sender and receiver to share secret keys via some secure channel is eliminated, all communications involve only public keys, and no private keys is ever transmitted or shared. Because conventional cryptography was once the only available means for relaying secret information. the expense of secure channels and key distribution relegated its use only to those who could afford it .such as governments and large bank(or small children with secret decoder rings)

Public key encryptions is the technological revolution that provides strong cryptography to the adult masses. Remember the courier with the locked briefcase handcuffed to his wrist? Public-key encryption puts him out of business (probably to his relief)

How Decryption works

Decryption works in the reves.The recipient's copy of Data Encryption Compression Simulator uses his or her private key to receiver the temporary session key, which Data Encryption Simulator then uses to decrypt the conventionally encrypted cipher text.



The combination of the two encryption methods combines the convenience of public key encryption with the speed of conventional encryption. Conventional encryption is about 1,000 times faster then public key encryption .Public key encryption in turn provides a solution to key distribution and data transmission issues. Used together, performance and key distribution are improved without any sacrifice in security.

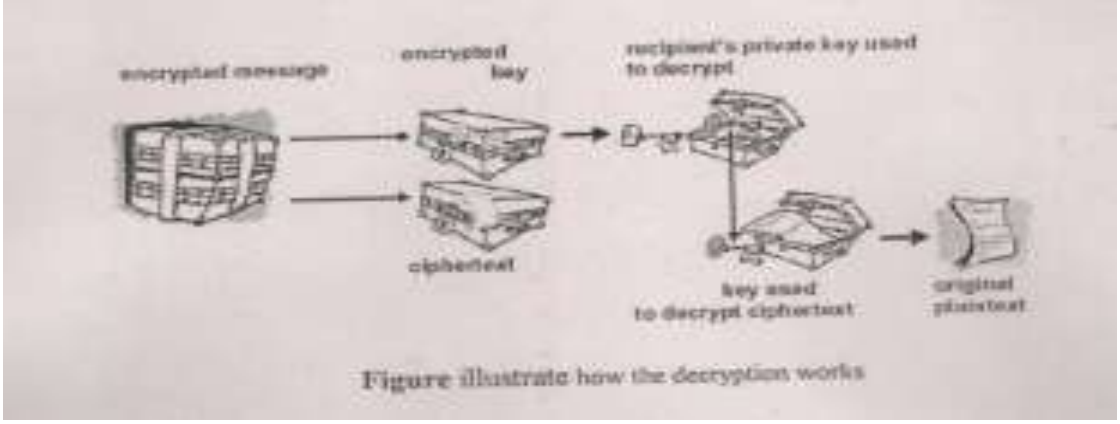
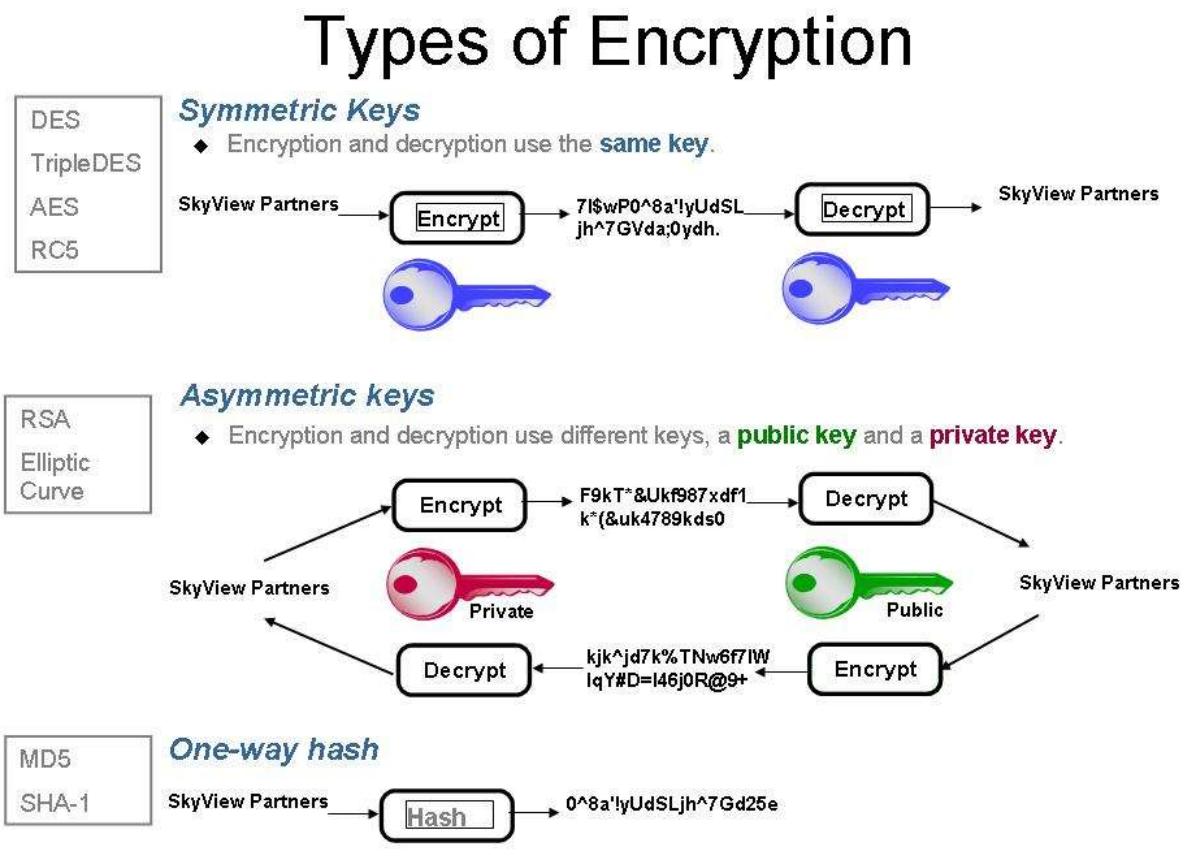


Figure illustrate how the decryption works



You'll need to consider these three types of encryption algorithms for your encryption project. (Click image to enlarge.)

- *Symmetric key algorithms* use the same key to both encrypt and decrypt data. They are generally faster than asymmetric key algorithms and are often used to encrypt large blocks of data. Algorithms you'll hear mentioned include DES, TripleDES, RC5, and AES (the standard for the U.S. government). Either TripleDES or AES is required when encrypting stored credit card numbers.
- *Asymmetric key algorithms* require two encryption keys: one to encrypt the data and the other to decrypt the data. It doesn't matter which one does which function; it's just that you can't use only one key to both encrypt and decrypt data. These algorithms are typically used for authentication; for example, they're used during an SSL or VPN "handshake" to verify that the client knows the server to which it's trying to connect. Once the verification is complete, a "session key" is exchanged by the client and server, and symmetric key encryption is used for the actual data transmission. Asymmetric key algorithm types include RSA and Elliptic Curve.
- *Hash algorithms* produce a result that is not decryptable. They are typically used for comparing two values to ensure they're the same. For example, digital signature verification uses a hash function to ensure that the signed document has not been altered. Because a hashed value cannot return the original value, this method provides the most secure form of storing data and is the most appropriate method when the original (cleartext) form is not required. MD5 and SHA-1 are two examples of hash algorithms.

You'll need to consider these three types of encryption algorithms for your encryption project. (Click image to enlarge.)

- *Symmetric key algorithms* use the same key to both encrypt and decrypt data. They are generally faster than asymmetric key algorithms and are often used to encrypt large blocks of data. Algorithms you'll hear mentioned include DES, TripleDES, RC5, and AES (the standard for the U.S. government). Either TripleDES or AES is required when encrypting stored credit card numbers.

- *Asymmetric key algorithms* require two encryption keys: one to encrypt the data and the other to decrypt the data. It doesn't matter which one does which function; it's just that you can't use only one key to both encrypt and decrypt data. These algorithms are typically used for authentication; for example, they're used during an SSL or VPN "handshake" to verify that the client knows the server to which it's trying to connect. Once the verification is complete, a "session key" is exchanged by the client and server, and symmetric key encryption is used for the actual data transmission. Asymmetric key algorithm types include RSA and Elliptic Curve.
- *Hash algorithms* produce a result that is not decryptable. They are typically used for comparing two values to ensure they're the same. For example, digital signature verification uses a hash function to ensure that the signed document has not been altered. Because a hashed value cannot return the original value, this method provides the most secure form of storing data and is the most appropriate method when the original (cleartext) form is not required. MD5 and SHA-1 are two examples of hash algorithms.

NON FUNCTIONAL REQUIREMNTS

SOFTWARES TOOLS

- .Net Frame Work
- VB.Net
- Microsoft Visual Basic
- SMTP SERVER

OPERATYING STYSTEM

Windows 2000,Windows XP

Use Case Diagram(s):

Use case diagrams displays the relationship among the actors and use cases.

Class Diagram

Models class structure and contents using design celemeents such a as classes packages and objects. It also displays relationships such as containment, inheritance, association and others.

Interaction Diagrams
Sequence Diagram displays the time sequence of the objects participating in the interaction. This consists of the vertical dimension (time)and horizontal dimension (different objects)

Collaboration Diagram

Displays an interaction organized around the objects and their links to one another. Numbers are used to show the sequence of messages.

State Diagram
Displays the sequences of states that an object of an interactions goes through during its life in response stimuli, together with its responses and actions.

Activity Diagram

Displays a special state diagram where most of the states are action states and most of the transition is triggered by completion of the actions in the source states. This diagram focuses on flows driven by internal processing.

Physical Diagram

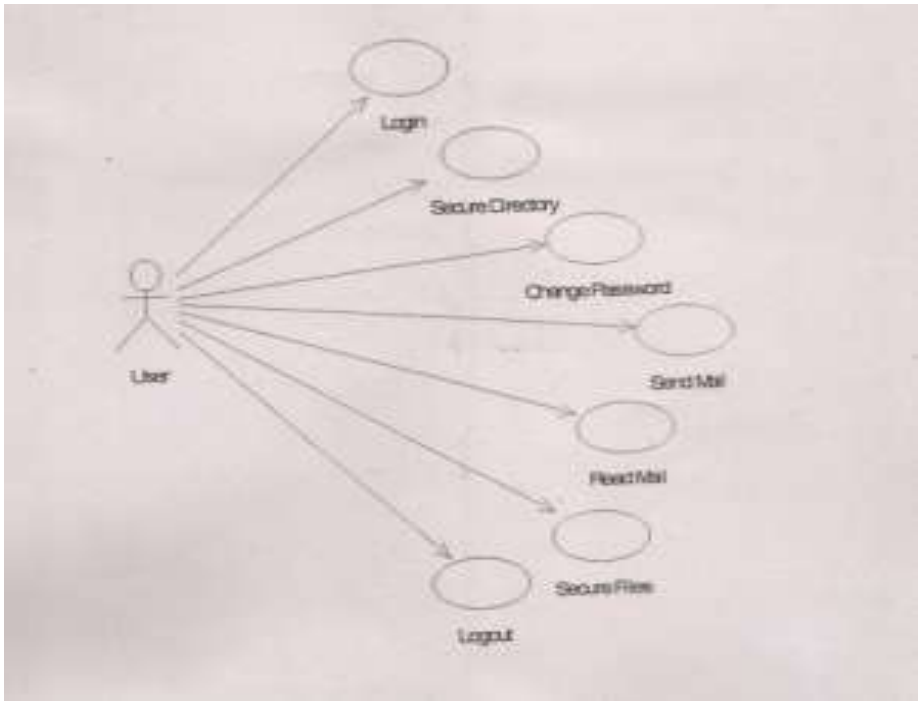
Component Diagram

Displays the high level packaged structure of the code itself. Dependencies among components are shown, including source code components, binary code components, and executable components. Some components exist at compile time, at run times well as at more than one time.

Deployment Diagram

Displays the configuration of run time processing elements and the software components processes, and objects that live on them. Software instances represent run-time manifestations of code units.

User case diagrams



ADMINISRATOR USE CASE

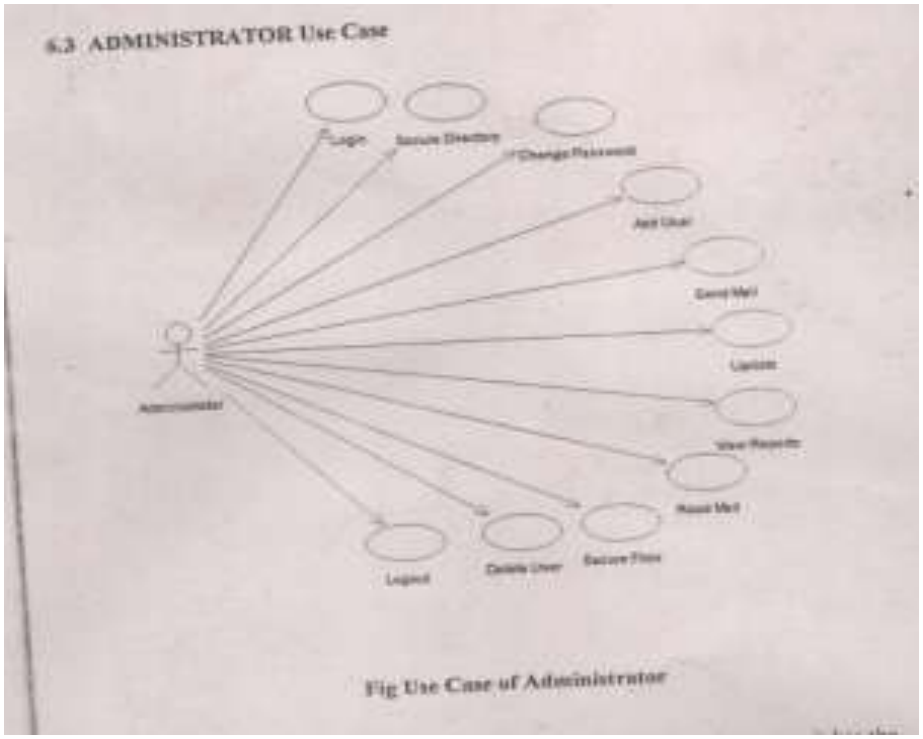


Fig Use Case of Administrator

Usage Scenarios:

<Provide here the usage scenarios of all use cases in table format explaining Use Case title, Use Case Id, Actions, Description, Alternative Paths, Pre and Post Conditions, Author, Exceptions. You are supposed to provide a usage scenario for each of use case shown in use case diagram>

There are following actors in the Data Encryption Compression Simulator

Actors Discription

User

Administrator

User

The user is the person that interact with the system to do the particular jobs of the software whichre are allowed to him.

The user can do following jobs

Creating message

Ending email message

Read email message

Secure file

Secure Directory

ADMINISTRATOR

The administrator is the user which has the maximum privileges. He can create new users, Administrator. He can delete or update the authorities of the existing user of the working. The administrator has also the authority of tall the works which are allowed to the common use. The administrator has the following authorities move given to him.

Add User
Delete User
Update user Authority
See Reports

USE CASE DISCRIPTION

Follwoin system use cased have been identified.

Login
Logout
Change Password
Add User
Delete User
Update User Authority
Send Email
Read Mail Secure Directory
Secure Text File

Login

Use Case	Logine
Actors	User,Administrator
Type	Primary and Essential
Description	Different users are allowed to use the system after providing Login name and password

Typical Course of Events

Actor Action		System Action	
1	This use case begin when user open Main page of application		
1	User choose the Login Type 1.User 2.Adminisrator	2	Validates Input
1	User provides the Login Name and password	3	Checks for existence so the login name and password in the user record. Upon successful redirect to the page with it authority.

Log Out

Use Case	Logout
Actors	To finish the work and close the usere windows
Type	Primary
Description	A user can not work any thing after the logout

Typical Course of Events

Actor Action		System Action	
1	User choose logout	2	It invalidates the user working. It redirect to the start page.

Change Password

Use case	Change Password
Actors	User, Administrator
Purpose	To change the Login Password
Type	Primary and essential

Typical Course of Events

Actor Action		System Action	
1	User choose change password	2	System prompts to the selected page. it prompts for the old and new password.
3	User provide the old and new password	4	System prompts the message if required password is changed after the old password is verified.

Add New User

Use Case	Add User
Actors	Administrator
Purpose	To add new person who can interact with the system
Type	Primary and essential
Description	This use case deals to add new user who can interact with the system.

Typical Course of Events

Actor Action		System Action	
1	This use case begins when administrator click the option: a Add new user b Start First Time	2	System redirect to the selected page.
3	On the selection of add user administrator enters the data of the new user	4	The user is added to the user list

Delete User

Use Case	Delete User
Actors	Administrator
Purpose	To delete person who interact with the system
Type	Primary and essential
Description	This use case deals to delete user who interact with the system.

Typical Course of Events

Actor Action		System Action	
1	This use case begins when administrator click the option Delete User	2	System redirect to the selected page.
3	On the selection of the delete user administrator enters the date of the user to whom he want to be deleting.	4	The user is added to the deleted list. And removed from the user list if the administrator

Update User Authority

Use Case	Update User Authority
Actors	Administrator
Purpose	To update person authority who interact with the system
Type	Primary and essential
Description	This use case deals to update user who interact with the system.

Typical Course of Events

Actor Action		System Action	
1	This use case begins when administrator click the option Update User	2	System redirect to the selected page.
3	On the selection of update user administrator enters the data of the user to whom he want to be update.	4	The user authority is updated to the user list. if the administrator is authorized to this job.

SEND E-MAIL

Use Case	Send Email
Actors	Authorized person may be user or administrator
Purpose	To send an email to user
Type	Primary and essential
Description	This use case deals with creating a secure message and sending it to the user.

Typical Course of Events

Actor Action		System Action	
1	This use case with the user click the option send message if he is authorized to it.	2	System open the Email Creation form
3	The user enter the following information. 1 The address of the person to whom the given message is to be sent 2 Email text or the secret information to be sent 3.Algorithm selection	4	System encrypt to the user text into the chipher text.
5	User load an data on which the message is to be set	6	Loads the data in encryption form
7	The user clicks the set message to the picture	8	System set the message on the picture and the second chippers is completed an the system says the user to save this image so this might be available

			for the attachment to he message.
9	Users add more attachment if he wants and clicks send.		System writes the mail on the disk queue and then forwards it to the particular user.

READ MESSAGE

Use Case	Read Email
Actors	Authorized person may be user or administrator
Purpose	To read an email to user
Type	Primary and essential
Description	This use case deals with creating a secure message and converting it to the plan text.

Typical Course of Events

Actor Action		System Action	
1	This use case with the user click the option Read message if he is authorized to it.	2	System open the Email Read form
3	The user login to his account in the web browser Download it s mail message	4	System inter act with internet and enables user to down load his emails.
5	User load an data on which the message is to be set	6	Loads the data in encryption form
7	The user clicks the read message to the picture	8	System set the message on the picture and the second verifies the user password. Gives the chipper text
9	The user click decrypt message	10	System verifies the user and then give

			the actual message to the user.

SECURE FILE

Use Case	Secure File
Actors	Authorized person may be user or administrator
Purpose	To secure the text or doc type documents on the hard disk of the user and restricting them from other users by the pass phrase.
Type	Primary and essential
Description	This use case deals with week cryptography.

Typical Course of Events

Actor Action		System Action	
1	This use case begins with the user click the option secure.	2	System open the text file security from
3	The user enter the following information 1.the address of the person to whom the given message is to be sent 2. Email text or the secret information to be sent 3.Algorithm slections	4	System encrypt to the user text into the chipper text.
5	User load an data on which the message is to be set	6	Loads the data in picture box
7	The user clicks the set message to the picture	8	System set the message on the picture and the second chipher is completed an the system says the user to save this image so this might be available

			for the attachment to the message.
9	User add more attachment file he wants and clicks send.	10	System writes the email on the disk queue and then forwards it to the particular user.

SECURE DIRECTORY

Use Case	Secure user directory
Actors	Any registered authorized user.
Purpose	To secure crypto system user directories from un authorized users.
Type	Primary and essential
Description	Use select the directory and set the password on the directory to which he wants to secure from the other users.

Typical Course of Events

Actor Action		System Action	
1	This use case begins with the user clicks the option secure directory	2	System open the security form.
3	The user selects the directory and then try to set the password on it	4	System check the given file is not on the desktop or the window directory and then set the security lock on the folder.

The process of unsecured is reverse of it the user select the directory and then gives the password of the particular directory.