UTIU
UNIVERSITÀ TELEMATICA
INTERNAZIONALE UNINETTUNO

جامعة حلوان
Helwan University

INFORMATION AND COMMUNICATION TECHNOLOGIES ENGINEERING

BACHELOR THESIS

DEVELOPMENT OF SAFE AND SECURE INTERNATIONAL FORMS USING DIGITAL BARCODES

Supervisors

Prof. Sameh A.Salem
Prof. Claudio Fornaro
Dr. Sandro Fontana

Students

Ali Salah Ibrahim
Andrew Reda Ghazal
Bishoy Raouf Abdalla Saleeb
Mohammed bakr Rabeeh
Tasnim Yousri Fouad

3 August 2011

Abstract

Every day, people make hundreds of deals. They buy, sell, rent, take loans, offer services and pay for others. So, there are millions of deals a day, that need documentation and these documents need to be secured from being corrupted or forged. However, seals, stamps, finger prints and signatures are old fashioned and can be easily changed, forged or even stolen, making the deal's documents easy to be corrupted. There is a need for nontraditional methodology to secure these deals out of corruption. Besides, need to apply computer programming and new management concepts for the optimal and safe dealing operations.

This work focuses on the application of QR barcodes to represent the data in the document in a secure way that cannot be corrupted or faked. It discusses a solution that solves many of the security concerns related to documents and the safety of their contents. In addition to programming software is designed and built for the application of Barcode scanner and mobile checker. The software is now being established for safe and secure international forms using digital barcodes.

Contents

Figure List

Task Distribution

Ali Salah

- Designing a bilingual program that will run different processes according to users' needs.
- Building a portable barcode checker that decrypts the data read from the barcode and implementing it in a mobile phone (with camera).

Andrew Ghazal

- Creating the QR Barcode from the data in the contract (Encoding) and extracting the original data in the contract from the barcode (Decoding).
- Backup and restore of critical files needed to run the program.

Bishoy Raouf

- Integration of the project parts created by the group members altogether and interfacing modules with each other.
- Overall system analysis and discussing the solution with prospected users.
- Creating HTML document forms and managing data entered by the user into them using java.

Mohammed Bakr

- Building a database (Users & documents) in order to organize the encoding process and connecting them to Java making it easier for the user.
- Testing the whole system in different phases to make sure it works properly.
- Writing the program user manual for guidance.

Tasnim Yousri

- Cryptography discussions and writing its codes.
- Building the codes for scanning documents to extract barcode by default in the decoding phase.

Chapter One
Introduction

The very beginning

People started documenting their deals using a third party who played the witness role. This solution was a great success until business got bigger and more serious.
Later, some other solutions appeared (locked boxes, finger prints, stamps, seals, signatures…etc).

21st century

Today, it's the paper documents era where millions of paper documents are released every day. Paper documents include checks, contracts, certificates, bills of exchange, promissory notes, formal letters and many others. Documents are also used in many purposes other than business.

We mention for example: Business, education, military, health, and economy. Some of these documents are supposed to be confidential, others need authentication security and some just need integrity security. Since these documents are flying everywhere, security is difficult to guarantee.

Problem Statement and Objectives

However, seals, stamps, finger prints and signatures are being considered old fashioned and can be easily changed, forged or even stolen, making the deal's documents easy to be corrupted. So, there is a need for nontraditional methodology to secure these deals out of corruption. Besides, need to apply computer programming and new management concepts for the optimal and safe dealing operations.

Methods

Signatures and stamps are hundreds of years old, and they are easy to forge, change or steal. So we decided to find another solution that is more secure. A solution that guarantees keeping your documents forgery free, away from hackers who want to view the contents and at the same time makes sure that the documents you send to an entity are delivered to the very same intended destination.

Initial visions

The following are some ideas we had in our start

First: *Computerizing the whole process.*

Simply, when a contract between two parties is being written, we put a computer (server) in the middle; the computer saves the contract and sends a copy to each party's e-mail. The problem is that any hacker that could access the server can put his own touch actively attacking all security targets and our mission is to avoid this.

Second: *Encryption.*

For ordinary barcodes; how the barcode is generated is a function of the data it hides, which can make a way for attacking with a barcode reader.
In our situation ,this is not the case ,the data which the barcode handles is not is not the data filled in the contract but rather it is encrypted via two phases, each phase of them is made of the latest state of art un-cracked algorithms.

Third: *Using a barcode.*

After the contract is written, it is scanned and then a barcode is generated and becomes the reference that both parties refer to.

Our project comes to life

We decided to mix many of the ideas we had and edit some details to find an ultimate solution that's difficult to attack in any of its stages.

The solution presents a precise process that is computerized, strong and implements an encrypted unique barcode to the document in contrast, raising safety levels for the data contained, and leaving active and passive attackers helpless.

Authentication problems are also solved while using the solution.

Our solution

1- Secure exchange of documents between dealing parties.
2- Any single simple change in documents after approval is spotted precisely.
3- Full authentication between dealing parties.
4- Parties do not have to sign at the same time making no force on any party to sign at specific time (this also helps when there is time zone difference).
5- Parties do not have to be online while dealing with documents.
6- There is no original copy (parties can print a copy and use anytime).
7- Introduces a user friendly program that helps the parties through the signing process.
8- Backup for the database and document forms used with the program.
9- A portable mobile checker that facilitates viewing decrypted original data instantaneously.

Chapter Two
Literature Reviews

In this chapter, we discuss all of our project's columns.
We discuss the tools that were homogeneously put together, in order to create the final program.

They were carefully chosen by the group of our project.
Every individual had his own opinions and we had to go through several long discussions to reach a satisfactory result.

Java language is used for building the main program that operates the whole project (encoding/decoding, encryption/decryption algorithms…etc).

The barcodes used are the **QR barcodes**, a unique two dimensions array that represents the output of the encryption algorithms (cipher text).

For showing the documents forms to the users we used **HTML**, and to embed the data entered by the user to the HTML into the main program written in Java we chose **JavaScript**.

The encryption process is held twice using the **RSA** encryption algorithm to ensure both authentication and confidentiality of the contents of the document in contrast. (RSA algorithm will be discussed further on)

AES encryption algorithm is also used in the encryption process giving the document more attacking resistance thus more security. (AES algorithm will be discussed further on).

For the databases used in the project to store user's information, contract forms and backing up data we used the **SQL** database language.

Here we start the discussion

QR code (Quick Response code)

High Capacity Encoding of Data

While conventional bar codes are capable of storing a maximum of approximately 20 digits, QR Code is capable of handling *several dozen to several hundred times more information.*

QR Code is capable of handling all types of data, such as numeric and alphabetic characters, Kanji, Kana, Hiragana, symbols, binary, and control codes. Up to 7,089 characters can be encoded in one symbol.

QR Code Data capacity	
Numeric only	Max. 7,089 characters
Alphanumeric	Max. 4,296 characters
Binary (8 bits)	Max. 2,953 bytes
Kanji, full-width Kana	Max. 1,817 characters

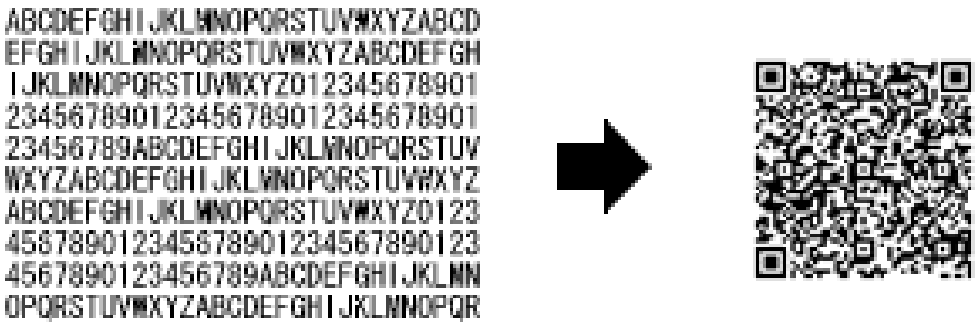


Fig 2.1

(A QR Code symbol of this size can encode 300 alphanumeric characters)

Small Printout Size

Since QR Code carries information both horizontally and vertically, QR Code is capable of encoding the same amount of data in approximately one-tenth the space of a traditional bar code.

Fig 2.2

Kanji and Kana Capability

As a symology developed in Japan, QR Code is capable of encoding JIS Level 1 and Level 2 kanji character set.
In case of Japanese, one full-width Kana or Kanji character is efficiently encoded in 13 bits, allowing *QR Code to hold more than 20% data than other 2D symbologies.*

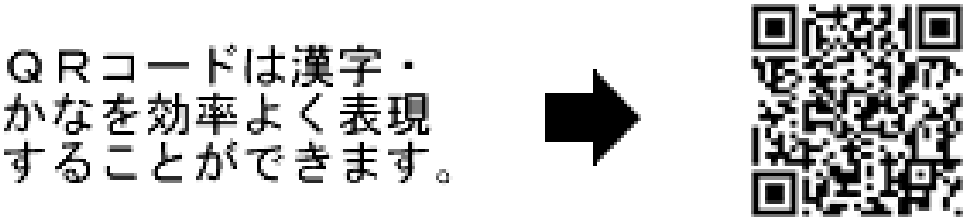


Fig 2.3

Dirt and Damage Resistant

QR Code has error correction capability. *Data can be restored* even if the symbol is partially dirty or damaged. A maximum 30% of codeword can be restored.

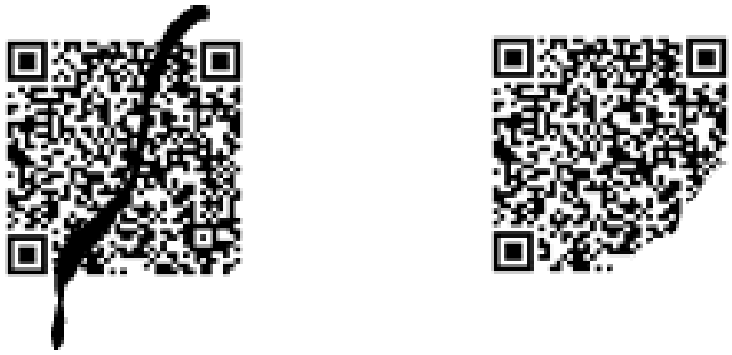


Fig 2.4

Readable from any direction in 360°

QR Code is capable of 360 degree (Omni-directional), high speed reading. QR Code accomplishes this task through position detection patterns located at the three corners of the symbol. These *position detection patterns* guarantee stable high-speed reading, circumventing the negative effects of background interference.

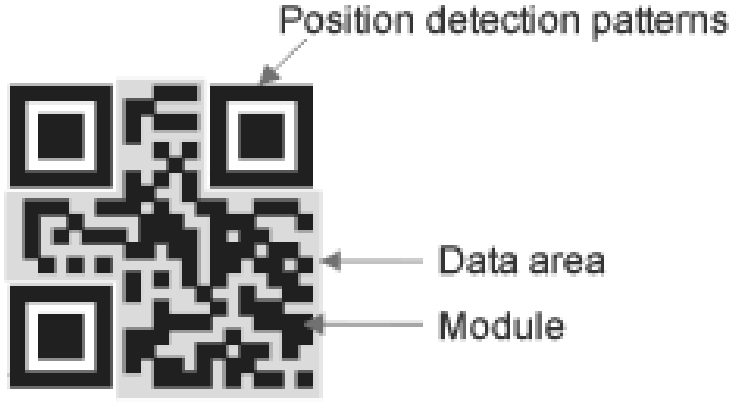


Fig 2.5

Structured Append Feature

QR Code can be divided into multiple data areas. Conversely, information stored in multiple QR Code symbols can be reconstructed as single data symbols.
One data symbol can be divided into up to 16 symbols, allowing *printing in a narrow area.*

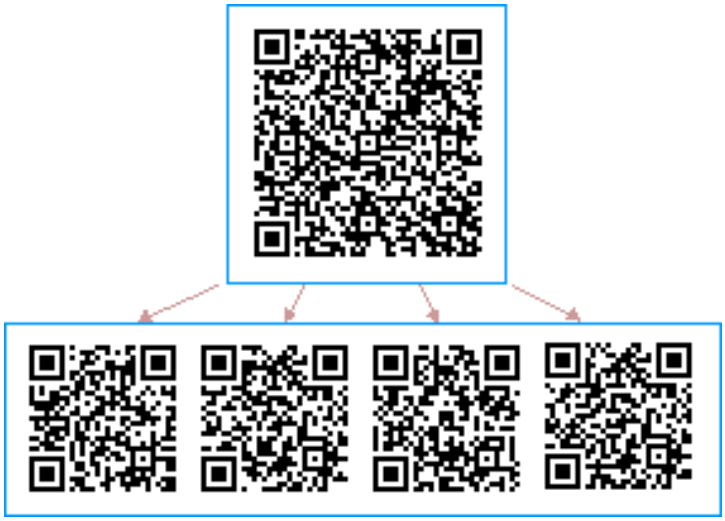


Fig 2.6

The same data can be read either from the upper symbol or the lower four symbols.

Bar code to 2D Code

Bar codes have become widely popular because of their reading speed, accuracy, and superior functionality characteristics.

As bar codes became popular and their convenience universally recognized, the market began to call for codes capable of storing more information, more character types, and that could be printed in a smaller space.

As a result, various efforts were made to increase the amount of information stored by bar codes, such as increasing the number of bar code digits or layout multiple bar codes.

However, these improvements also caused problems such as enlarging the bar code area, complicating reading operations, and increasing printing cost.

2D Code emerged in response to these needs and problems

Multiple bar code layouts

2D Code with stacked bar codes (stacked bar code type)

2D Code (matrix type)

2D Code is also progressing from the stacked bar code method (that stacks bar codes), to the increased information density matrix method.

About QR Code

Fig 2.7

QR Code (2D Code) contains *information in both the vertical and horizontal directions*, whereas a bar code contains data in one direction only. QR Code holds a considerably greater volume of information than a bar code

Typical 2D Code

In addition to QR Code, some other kinds of 2D Code have been developed. Below is a table of typical 2D Code and their features.

		QR Code	PDF417	DataMatrix	Maxi Code
					
Developer(country)		DENSO(Japan)	Symbol Technologies (USA)	RVSI Acuity CiMatrix (USA)	UPS (USA)
Type		Matrix	Stacked Bar Code	Matrix	Matrix
Data capacity	Numeric	7,089	2,710	3,116	138
	Alphanumeric	4,296	1,850	2,355	93
	Binary	2,953	1,018	1,556	
	Kanji	1,817	554	778	
Main features		Large capacity, small printout size High speed scan	Large capacity	Small printout size	High speed scan
Standardization		AIM International JIS ISO	AIM International ISO	AIM International ISO	AIM International ISO

QR Code Outline Specification

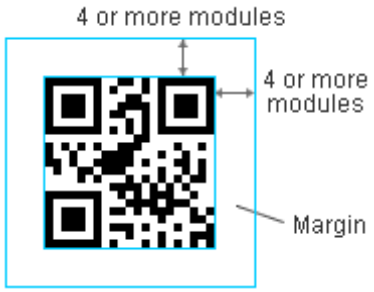
Symbol size	21 × 21 - 177 × 177 modules (size grows by 4 modules/side)	
Type & Amount of Data (Mixed use is possible.)	Numeric	Max. 7,089 characters
	Alphanumeric	Max. 4,296 characters
	8-bit bytes (binary)	Max. 2,953 characters
	Kanji	Max. 1,817 characters
Error correction (data restoration)	Level L	Approx. 7% of codewords can be restored.
	Level M	Approx. 15% of codewords can be restored.
	Level Q	Approx. 25% of codewords can be restored.
	Level H	Approx. 30% of codewords can be restored.
Structured append	Max. 16 symbols (printing in a narrow area etc.)	

Securing Margin

When the symbol version and module size are determined, the size of the QR Code symbol is determined. The QR Code symbol area requires a margin or "quiet zone" around it to be used.

The margin is a clear area around a symbol where nothing is printed. QR Code requires a four-module wide margin at all sides of a symbol.

Margin of QR Code



Four-module wide margin is required around a symbol.

Example of Calculating QR Code Area

Below is an example of calculating the total QR Code area including margin.
(Example) Creating QR Code to encode 50 alphanumeric characters

1. Specify the error correction level as the standard "M".
2. Obtain a version from the Version and maximum data capacity table (find the intersection of alphanumeric characters and Level M). → Version 3 capable of storing 50 or more characters. (Version 2 with Level M holds only 38 characters.)
3. Use a printer with 400 dpi resolution. → 0.254 mm when printed with 4-dot configuration.(Equation: 25.4 mm/inch ÷ 400 dpi × 4 dots/module = 0.254 mm/module)
4. 4. Version 3 = 29 modules, therefore, the size of QR Code is 29 modules × 0.254 mm/module = 7.366 mm.
5. Secure a four-module wide margin. 7.366mm + 0.254mm/module × 8 modules =*9.398mm*

In other words, the required QR Code area is 9.398mm^2.

If QR Code Area gets Too Large

If the QR Code area obtained in the process above does not fit the printing space, consider the following three points.

1. Decrease the symbol version.
2. Make the module size smaller.
3. Split the QR Code symbol.

Symbol Version

The symbol versions of QR Code range from Version 1 to Version 40. Each version has a different module configuration or number of modules. (The module refers to the black and white dots that make up QR Code.)

"Module configuration" refers to the number of modules contained in a symbol, commencing with Version 1 (21 × 21 modules) up to Version 40 (177 × 177 modules). Each higher version number comprises 4 additional modules per side.

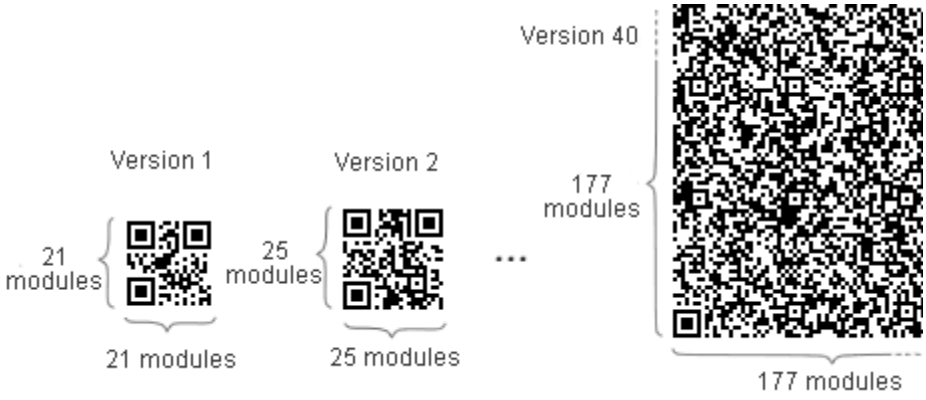


Fig 2.8

Each QR Code symbol version has the maximum data capacity according to the amount of data, character type and error correction level. Check the maximum data capacity for each version.
In other words, as the amount of data increases, more modules are required to comprise QR Code, resulting in larger QR Code symbols.

Error Correction

QR Code has error correction capability to restore data if the code is dirty or damaged. Four error correction levels are available for users to choose according to the operating environment. Raising this level improves error correction capability but also increases the amount of data QR Code size.

To select error correction level, various factors such as the operating environment and QR Code size need to be considered.

Level Q or H may be selected for factory environment where QR Code get dirty, whereas Level L may be selected for clean environment with the large amount of data. Typically, Level M (15%) is most frequently selected.

QR Code Error Correction Capability*	
Level L	Approx.7%
Level M	Approx. 15%
Level Q	Approx. 25%
Level H	Approx. 30%

*Data restoration rate for total codewords (codeword is a unit that constructs the data area. One codeword of QR Code is equal to 8 bits.)

Error Correction Feature

The QR Code error correction feature is implemented by adding a *Reed-Solomon Code*[*] to the original data.

The error correction capability depends on the amount of data to be corrected. For example, if there are *100 codewords of QR Code* to be encoded,*50 of which need to be corrected, 100 codewords* of Reed-Solomon Code are required, as Reed-Solomon Code *requires twice the amount of* codewords to be corrected.

In this case, the total codewords are 200, 50 of which can be corrected. Thus, the error correction rate for the total codewords is 25%. This corresponds to QR Code error correction Level Q.

In the example above, the error correction rate for QR Code codewords can be considered as 50%. However, it is not always the case that codewords of not Reed-Solomon Code but only QR Code are susceptible to dirt and damage. QR Code therefore represents its error correction rate as a ratio of the total codewords.

(*) Reed-Solomon Code is a mathematical error correction method used for music CDs etc. The technology was originally developed as a measure against communication noise for artificial satellites and planetary probes. It is capable of making a correction at the byte level, and is suitable for concentrated burst errors.

The importance of QR code in our project:

From all the previous information about the QR barcode we found that it is the most suitable choice for us because.

1. It holds a lot of data.
2. Its size is relatively small related to the other barcodes.
3. The error correction feature will help in restoring destroyed documents.

HTML

HTML is a markup language that is a set of markup tags that is used by HTML to describe web pages. It is not considered a programming language

HTML is the abbreviation of **H**yper **T**ext **M**arkup **L**anguage.
It is used to create the sample documents' forms used in the program.

The HTML form is opened in a web browser; we used Internet Explorer as our default browser.
The purpose of a web browser is to read HTML documents and display them as web pages. The browser does not display the HTML tags, but uses the tags to interpret the content of the page.

HTML has many advantages over its alternatives

- **Browser compatibility**
  Validated code ensures your document is compatible with the current browsers and future browsers, making it suitable for future expansion.
  The program also has a built in browser that's compatible with HTML.

- **Faster Loading**
  The document form has to load in under 10 seconds, or the user will lose interest. In our case it opens in much less time as the program works offline.

- **Easier to add, update and maintain our documents**
  Possibility to add a form for a new user or new company requirements any time.
  If the HTML code has no mistakes, then it is easier and faster to make changes to any document form. This means saving time when maintaining clients' document formats.

We use HTML for displaying our program forms like checks, certificates, contracts, etc in its typical formats, just like the original ones.

JavaScript

Have many advantages that made it our first choice.
JavaScript was designed to add interactivity to HTML pages and is usually embedded directly; it is a scripting language which is a lightweight programming language.

It is an object-oriented scripting language used to enable programmatic access to objects within both the client application and other applications. It does not need any license and works in all major browsers.

JavaScript is an interpreted language that does not need preliminary compilation to execute.

Java and JavaScript

Java and JavaScript are two completely different languages in both concept and design.
Java (developed by Sun Microsystems) is a powerful and much more complex programming language - in the same category as C and C++.

JavaScript can be used to store and retrieve information on the user's computer **We use JavaScript in fetching data entered by the users in the documents and passing it to our program to be executed in data encryption, and data encoding for producing the barcode that represents the data.**

JavaScript features

- **JavaScript gives HTML designers a programming tool** - HTML authors are normally not programmers, but JavaScript is a scripting language with a very simple syntax! Almost anyone can put small "snippets" of code into their HTML pages
- **JavaScript can read and write HTML elements -** A JavaScript can read and change the content of an HTML element

Here is the JavaScript code that fetches the data entered by users in HTML documents:

```
<script type="text/javascript">
	//This method writes the values in a file
	function WFile(Txt)
	{
		var ForAppending = 8;
		var TriStateFalse = 0;
		var ForWriting = 2;
		var fso = new ActiveXObject("Scripting.FileSystemObject");
		var newFile =
		fso.OpenTextFile("C://SUDB/Forms/Certificate_Data.txt",
		ForWriting, true, TriStateFalse);
		newFile.WriteLine(Txt);
		newFile.Close();
		window.close();
	}

	var theValues = new Array();

	//This method gets the values from the textboxes
	function getVariables()
	{
		var obj = document.getElementById('theOpen');//Gets the
		parent DIV
		var inputs = obj.getElementsByTagName('input');//Gets the
		inputs inside the parent DIV
		for(i=0;i<inputs.length-1;i++)
			{theValues[i] = inputs[i].value;}//Stores values in array
		displayValues();//Calls displayValues method below
	}
```

```
	//This method displays the values
	function displayValues()
	{
		obj2 = document.getElementById('displayThem');//Gets the
		DIV to display them in
		obj2.innerHTML = "";//Sets it so its empty
		for(i=0;i<theValues.length;i++)
		obj2.innerHTML += theValues[i] + "`";//Adds each value in
		theValues array to the DIV.
		WFile(obj2.innerHTML);
	}
</script>
```

Cryptography and encryption

Cryptography

It is the science of using mathematics (complex mathematical processes) to encrypt and decrypt information. Once the information has been encrypted, it can be stored on insecure media or transmitted on an insecure network (like the Internet) so that it cannot be read by anyone except the intended recipient.

Cryptography Goals

Authentication

This means that before sending and receiving data using the system, the receiver and sender identity should be verified.

Secrecy or Confidentiality

Usually this function (feature) is how most people identify a secure system. It means that only the authenticated people are able to interpret the message (date) content and no one else.

Integrity

Integrity means that the content of the communicated data is assured to be free from any type of modification between the end points (sender and receiver). The basic form of integrity is packet check sum in IPv4 packets.

Non-Repudiation

This function implies that neither the sender nor the receiver can falsely deny that they have sent a certain message.

Service Reliability and Availability

Since secure systems usually get attacked by intruders, which may affect their availability and type of service to their users. Such systems should provide a way to grant their users the quality of service they expect.

Encryption and decryption processes

Encryption is the process in which data (plaintext) is translated into something that appears to be random and meaningless (ciphertext). Decryption is the process in which the ciphertext is converted back to plaintext.

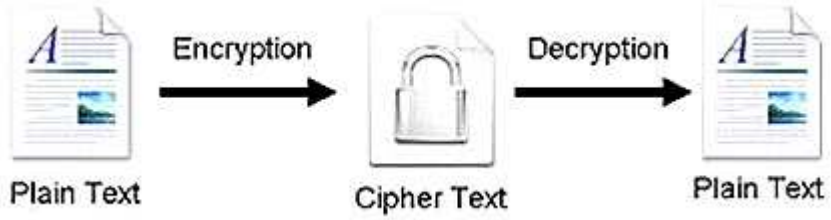


Fig 2.9

Encryption software is needed for

Communication and sending sensitive messages and files over the Internet for example. If you need to communicate or send information over the Internet you should encrypt it first.
With encryption and decryption software you can safely send sensitive messages and files on the web.
Need to safely store sensitive information on the computer and not to be hacked by others.

Encryption procedures are

Encryption program uses an encryption algorithm to encrypt and decrypt the data. Encryption algorithm creates specific strings of data used for encryption and keys that consist of long strings of bits. The more bits in the key, the more the number of possible combinations of binary numbers that makes the code more difficult to break. Then encryption algorithm scrambles data by combining the bits in the key with the data bits. In symmetric encryption, the same key is used to scramble (encrypt) and unscramble (decrypt) data. In asymmetric key encryption, two different keys are required - one for encryption and one for decryption.

Cryptanalysis types of attacks

A Fundamental rule

One must always assume that the attacker knows the methods for encryption and decryption; he is only looking for the keys.

Passive attack

The attacker only monitors the traffic attacking the confidentiality of the data

Active attack

The adversary attempts to alter the transmission attacking data integrity, confidentiality, and authentication.

Cryptanalysis

Rely on the details of the encryption algorithm plus perhaps some knowledge about the general characteristics of the plaintext –sometimes the plaintext is known and the key is being looked for.

Brute-force attack

Try every possible key on the ciphertext until an intelligible translation into a plaintext is obtained

A Figure that represents a basic situation in cryptography

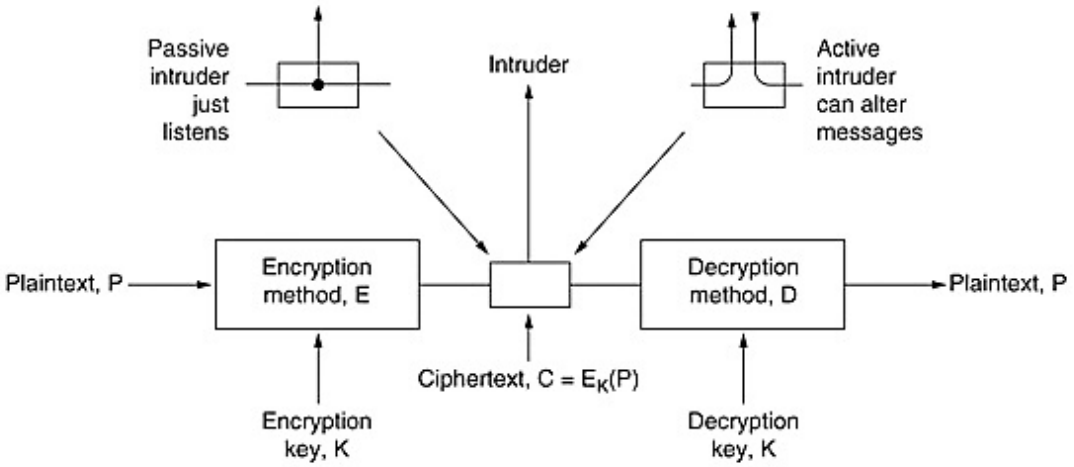


Fig 2.10

Cipher

The word “cipher” in former times meant “zero” and had the same origin: Middle French as cifre and Medieval Latin as cifra, from the Arabic word صفر “sifr”.
“Cipher” was later used for any decimal digit, even any number. There are many theories about how the word “cipher” may have come to mean “encoding”:
The Roman number system was cumbersome because there was no concept of zero. The concept of zero was very alien in medieval Europe. Cipher came to mean concealment of clear messages or encryption. It was concluded that the Arabic word for the digit zero, developed into the European technical term for encryption.

Fig. shows Caesar’s Ciphering example:

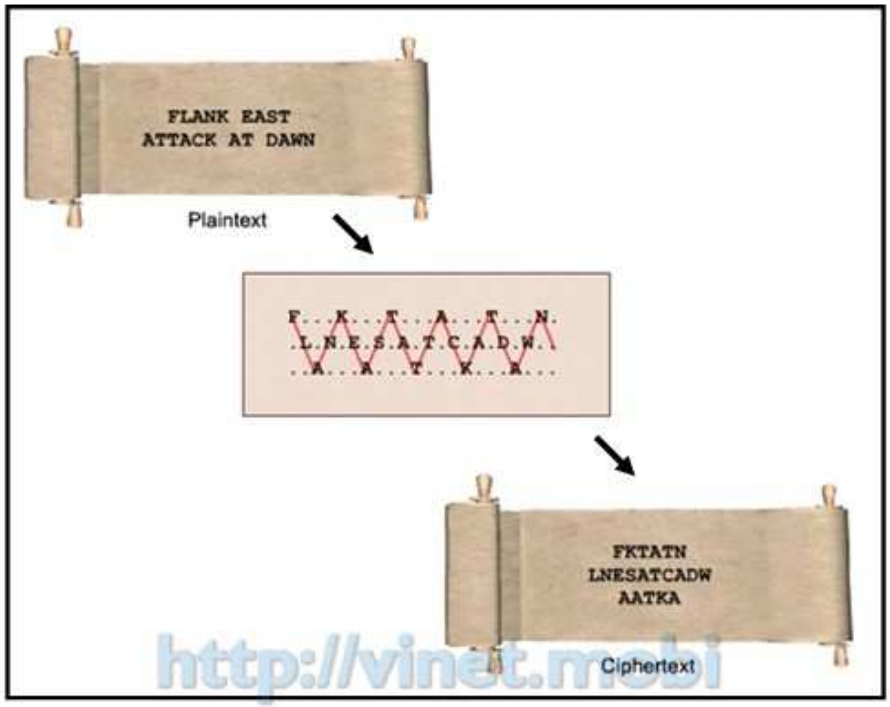


Fig 2.11

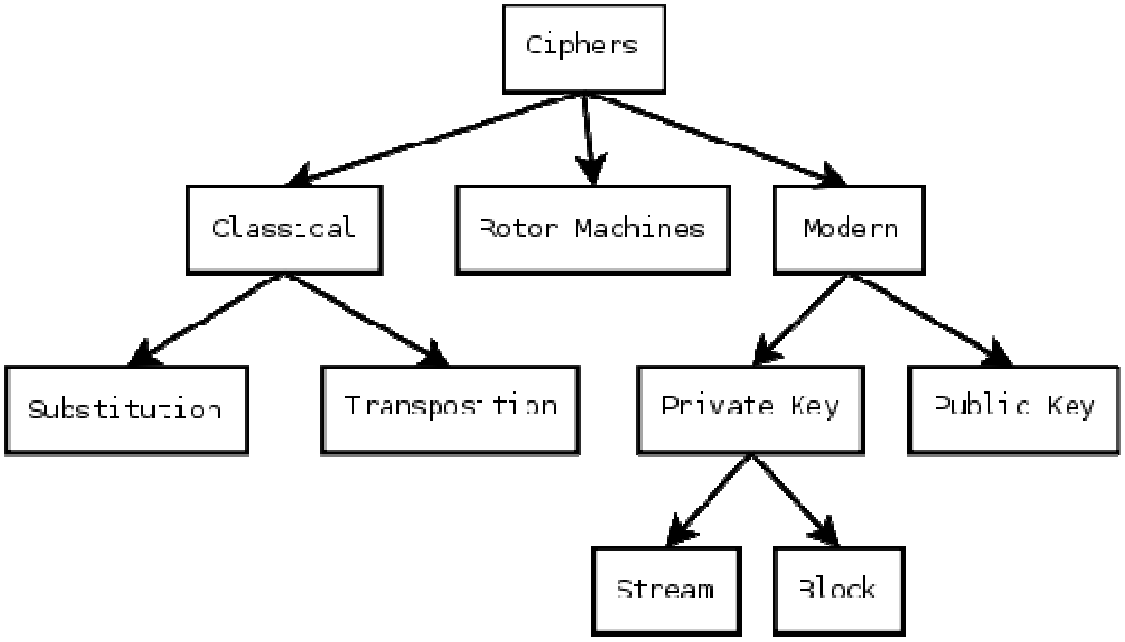


Fig 2.12

Features of Cryptographic Algorithms:

1. Level of protection

Some encryption techniques provide a virtually unbreakable barrier to information theft; others just require a determined attacker with moderate resources to be broken. One way to compare techniques on this level is to estimate how much CPU time would be required on a machine of a given processing speed to iterate through all the possible keys to the encoded data. For example, "A 128-bit XYZ cryptographic key requires 14.5 months of CPU time on an Acme 24-processor server to be broken." But other issues can affect the level of effort required to break the encrypted data, and make it difficult to objectively compare the security of encryption techniques. For example, if the attacker is not familiar with the format of the data being transmitted, and the data isn't easily interpreted on its own, then it may be tough to tell if an attempt to decode the data has worked or not.

2. Sophistication and complexity

Encryption techniques are usually based upon the mathematical properties of numbers and digital information. The mathematical theories employed in creating

encryption techniques vary in their complexity and sophistication; some require poring over many pages of mathematics and statistics journals to be fully understood, while others can be explained using basic concepts of algebra. The resources required to implement and to break a given encryption technique are usually a direct function of its complexity. It is always needed to trade off efficiency against security. If high throughput is a requirement for an application, you may be willing to use a less complex and less secure, cryptographic algorithm if it imposes significantly less overhead on your agents and vice-versa is true.

3. One-, two-, and many-way cryptography

Depending on the nature of the application, there may be situations where many parties need to be individually or mutually authenticated by one agent or more. A secure chat room for example, may require mutual authentication by all participating parties.

Most authentication schemes directly support one-way or two-way agent authentication.

Not many applications imply the concept of multi-way authenticated communications.

The developer's choice between one or two or multi-way authentication is based upon ,what the application he/she develops the software for and how many parties need to authorize each other.

4. Design issues

Modern cryptography involves the use of keys for data signing, encryption, and decryption. Some require the secure distribution such as private keys between parties, while others allow the parties to use public keys that can be broadcasted.

Other design issues involve the low-level implementation details of the algorithm. For an applet context example, the implementation of a cryptographic algorithm needs to be done completely in Java without any native method calls, so that we don't have to distribute native libraries to every agent that will talk to our agent.

5. Financial and legal issues

6. Certificates and authentication techniques

Encryption algorithms

Algorithms are two types

1) Symmetric
2) Asymmetric

Difference between symmetric and asymmetric encryption

Symmetric algorithms encrypt and decrypt with the same key.
Main advantages of symmetric algorithms are that they are the most unbreakable and secure algorithms and achieving massive high speed if compared to asymmetric ones.
Asymmetric algorithms encrypt and decrypt with different keys. Data is encrypted with a public key, and decrypted with a private key in case of encryption, while in case of Authentication data is encrypted with a private key, and decrypted with the public key. Asymmetric algorithms (also known as public-key algorithms) need at least a 3,000-bit key to achieve the same level of security of a 128-bit symmetric algorithm. Asymmetric algorithms are incredibly slow and it is impractical to use them to encrypt massive large amounts of data. Symmetric algorithms are about 1,000 times faster than asymmetric ones.
But Asymmetric algorithms glory is for the authentication needs, in a situation where symmetric algorithms cannot help.

Symmetric Encryption

Symmetric Encryption (also known as symmetric-key encryption, single-key encryption, one-key encryption and private key encryption) is a type of encryption where the same secret key is used to encrypt and decrypt information or there is a simple transform between the two keys.
A secret key can be a number or just a string of random letters. Secret key is applied to the information to change the content in a particular way. This might be as simple as shifting each letter by a number of places in the alphabet. Symmetric algorithms require that both the sender and the receiver know the secret key, so they can encrypt and decrypt all information. There are two types

of symmetric algorithms: Stream algorithms (Stream ciphers) and Block algorithms (Block ciphers).

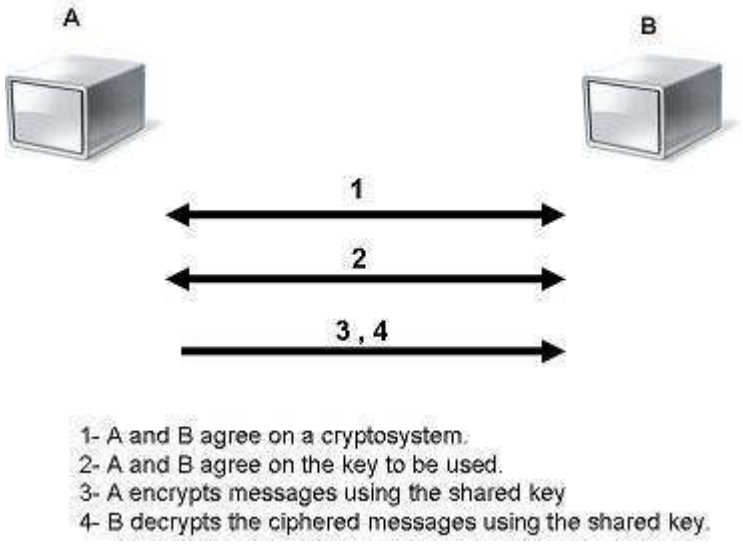


Fig 2.13

Fig of Symmetric encryption

Stream Cipher

The stream cipher is one of the simplest methods of encrypting data. When a stream cipher is employed, each bit of the data is sequentially encrypted using one bit of the key. A classic example of a stream cipher was the Vernam cipher used to encrypt teletype traffic. The crypto key for the Vernam cipher was stored on a loop of paper. As the teletype message was fed through the machine, one bit of the data would be combined with one bit of the key in order to produce the ciphertext. The recipient of the ciphertext would then reverse the process, using an identical loop of paper to decode the original message.

Some algorithms that use stream ciphers are:

RC4, Rabbit algorithm, SOBER, W7.

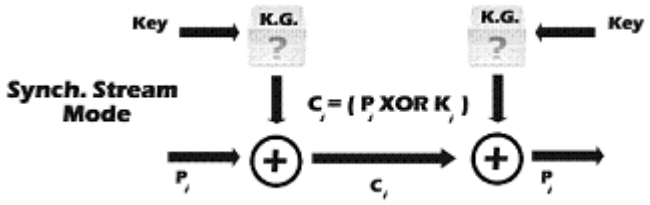


Fig 2.14

Stream Cipher (Simple Mode)

Block Cipher

Unlike stream ciphers, which encrypt every single bit, block ciphers are designed to encrypt data in chunks of a specific size. A block cipher specification will identify how much data should be encrypted on each pass (called a block) as well as what size key should be applied to each block. For example, the Data Encryption Standard (DES) specifies that DES encrypted data should be processed in 64-bit blocks using a 56-bit key. There are a number of different algorithms that can be used when processing block cipher encryption. The most basic is to simply take the data and break it up into blocks while applying the key to each. While this method is efficient, it can produce repetitive ciphertext. If two blocks of data contain exactly the same information, the two resulting blocks of ciphertext will be identical, as well. A cracker can use ciphertext which repeats in a nonrandom fashion to break the crypto key.

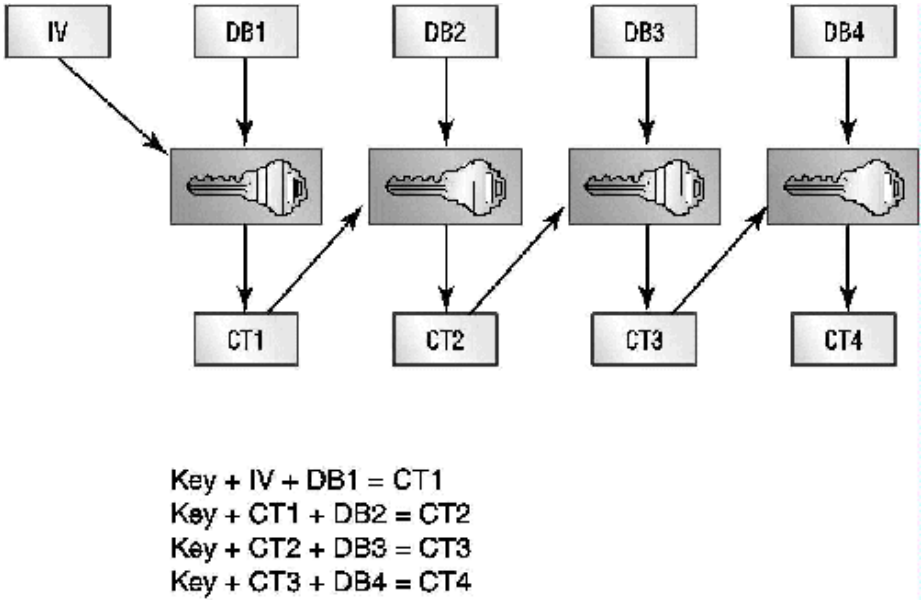


Fig 2.15
Block cipher encryption example

Famous Algorithms that use block cipher:

3DES, DES, AES, ARIA, Blow fish, RC5, CAST, CLEIA, CMAC, Mars, Rijndael, IDEA.

Asymmetric Encryption (Public Key Encryption)

Asymmetric encryption uses different keys for encryption and decryption. It is common to set up 'key-pairs' within a network so that each user has a public and private key. The public key is made available to everyone so that they can send messages, but the private key is only made available to the person it belongs to.

How Asymmetric Encryption (Public Key Encryption) works?

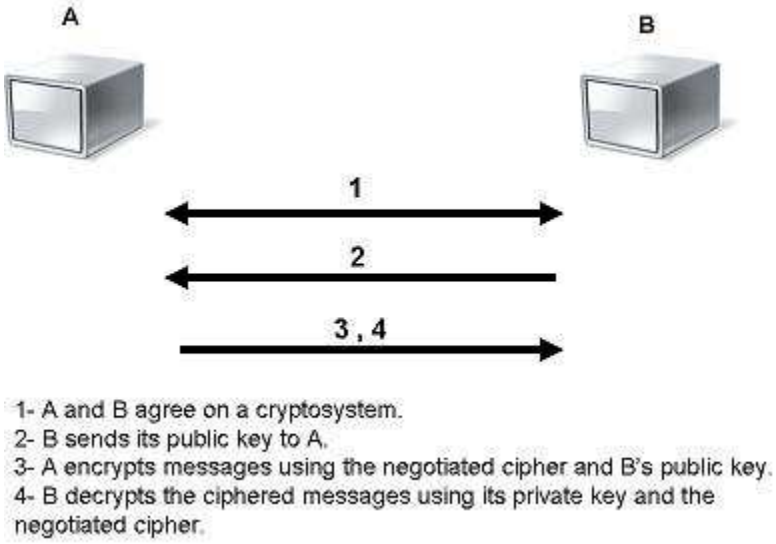


Fig 2.16
Asymmetric encryption case

Asymmetric encryption case
The sender and the recipient must have the same software. The sender does not need the recipient's password to use his or her public key to encrypt data. The recipient's other key is a private key that only he or she can use when decrypting the message and preserving it secure is his/her full duty ,since the private key assures that only the intended recipient can unscramble (decrypt) data intended for him or her.

For an authentication goal example, Jack has public key A and private key A, and Jill has public key B and private key B. Jack and Jill exchange their public keys. Once they have exchanged keys, they can communicate.

Jack wants to send an authorized message to Jill so, by using his private key he will scramble the message. Jill uses Jack's public key to unscramble it. In other words, since no one has Jack's private key but him, Jill will be sure that he is the sender.

Some asymmetric algorithms such as hash functions algorithms do not allow working in two directions or recovering the data again, and that is how digital signature works. So it's not applicable in the situations where you want to decrypt the sent data as in our software for example.

Now, the question is, what type of encryption did we choose? Is one stage enough to achieve both high security level and authentication?
Unfortunately the answer is NO, our choice was a combination of both Symmetric Encryption and Asymmetric Encryption together, but why?

To achieve all the five cryptography targets together, they are five intertwined technologies that help to insure that your data remains secure. The basic idea of our cryptography is based upon two phases of cryptography (unapproved then approved), discussed later on in the results chapter.

RSA for Asymmetric encryption

Public key algorithms are:

RSA

RSA is the best known asymmetric (public key) algorithm, named after its inventors. RSA uses public and private keys that are functions of a pair of large prime numbers. Its security is based on the difficulty of factoring large integers. The RSA algorithm can be used for both public key encryption and digital signatures. The keys are generated using random data. It is a multi-way algorithm.

DSA

The Digital Signature Algorithm (DSA) is a United States Federal Government standard or FIPS for digital signatures. It was proposed by NIST in August 1991 for use in their Digital Signature Standard (DSS). ONE-WAY algorithm and doesn't recover the real message data.

Hash functions

A hash function H is a transformation that takes a variable-size input m and returns a fixed-size message digest string, which is called the hash value h (that is, h = H(m)).

A hash function H is said to be one-way or hard to invert, where "hard to invert" means that given a hash value h, it is computationally infeasible to find some input x such that H(x) = h.
Just two digests at the sender's and receiver's parties are generated, should match or not only but plain data cannot be recovered again.

Conclusion

Due to the fact that in our application we INSIST on the recovery and unveiling the plain data again. Both hash functions and DSA were incompatible to our application, because they are one-way algorithms, in other words if there is a change made in the data before or after encryption, there is no way to know where the change is, and in which stage did it happen just all what you know is that there was a change (digital signatures un matching).
Consequently, strong un-cracked, multi-way RSA algorithm was our choice for our application's need.

RSA

The RSA algorithm can be used for both public key encryption and authentication. Its security is based on the difficulty of factoring large integers.
There are two keys should be given for every single user using this algorithm ,the first is called the private key which should receive the TOP SECRET treatment from the user ,because if he/she lost it his communications are no longer safe. The other is available to everyone and called the public key.

Keys are generated following this algorithm

Generate two large random primes, p and q, of approximately equal size such that their product n = pq is of the required bit length, e.g. 1024 bits as in our software.
Compute n = pq and (ϕ) phi = (p-1)(q-1).
Choose an integer e, 1 < e < phi, such that gcd(e, phi) = 1.
Compute the secret exponent d, 1 < d < phi, such that ed ? 1 (mod phi).
The public key is (n, e) and the private key is (n, d). Keep all the values d, p, q and phi secret.

Where:

n is known as the modulus ,e is known as the public exponent or encryption exponent or just the exponent ,d is known as the secret exponent or decryption exponent.

Encryption algorithm

Assume Adam wants to communicate to Bob, Adam does the following:
He should be having Bob's public key (n, e).
Real text message should be converted into its ASCII code representation before encryption to operate mathematically on it, where 'm' should satisfy the condition: 1<m<n, for the decryption stage to decrypt correctly, if not so: split the real message into smaller blocks each encrypted alone.
The ciphertext (c) is computed applying this mathematical relation: c = me mod n.
Adam sends 'c' to Bob.

Decryption algorithm

Recipient Bob does the following

Uses his private key (n, d) to compute the real message (m), where m = cd mod n.
The numerical ASCII 'm' is given to another code to return it back as a textual message.
How the authentication algorithm works?
For the sender 'A':

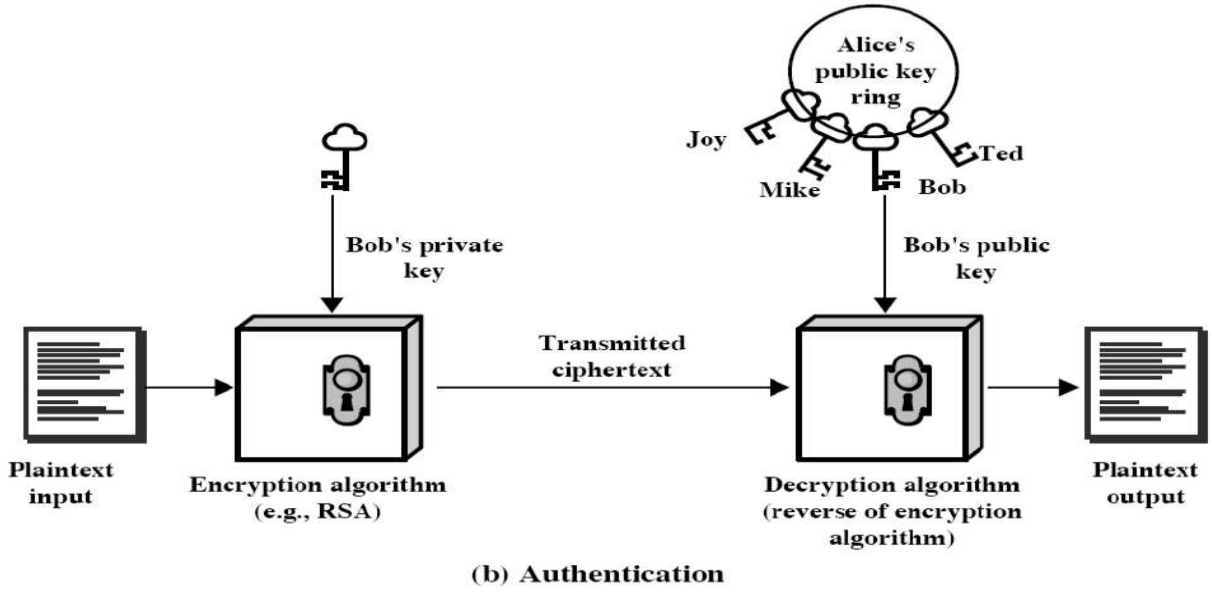
1.'A' wants to send an authenticated message to 'B'.
2.'A' encrypts the message using his/her private key.

Verifying the sender

For the receiver ‘B’:

3. When B receives the message, she/he decrypts it using A’s public key.

A’s public key is not a secret, thus people who have it can decrypt the message as well, but since no one has A’s private key but A , NO ONE can send or change in the contents of the message but ‘A’ which is called authentication ,where the receiver doesn’t care if someone will decrypt the contents with her/him or not ,rather she/he ONLY cares about: Is ‘A’ really the sender?



2.17

A figure showing the authentication of the sender of the message

Key length

When we talk about the key length of an RSA key, we are referring to the length of the modulus, n, in bits. A key length of 512 bits is now no longer considered that secure although cracking it is still not that trivial task but it is way faster than both key lengths 1024 and 2048.A key length of 2048 bits is considered to be very slow and generating a big cipher.
That is why 1024 bit key length was our choice to preserve a secure encryption with sort of accepted execution time, and also cipher's length not that big to subject the data multiple encryption stages, and to be compatible with the barcode encoder and decoder, that they can read ALL the cipher generated after all stages, and not be that massive for the QR barcode to handle.

Why AES as a symmetric encryption?

Different symmetric algorithms are:

DES

(Data Encryption Standard), was the first encryption standard to be recommended by NIST (National Institute of Standards and Technology). It is based on the IBM proposed algorithm called Lucifer. DES became a standard in 1974 .Since that time, many attacks and methods recorded that exploit the weaknesses of DES, which made it an INSECURE and an old already cracked block cipher.

3DES

As an enhancement of DES, and a way to try to face the fact that DES was cracked, the Triple DES encryption standard was proposed. In this standard the encryption method is similar to the one in original DES but applied 3 times to increase the encryption level. But it is a known fact that 3DES is slower than other block cipher methods.

Given the current state of technology, 3DES should be used with 3 keys

Drawbacks of 3DES

Relatively slow software implementations

DES was designed for 1970's hardware.

DES and 3DES use 64-bit blocks: for efficiency and security, larger blocks should be used.

Blowfish

It is one of the common public domain encryption algorithms provided by Bruce Schneier, It's the fastest symmetric algorithm but not the hardest to crack providing less security.

AES

(Advanced Encryption Standard), is the new encryption standard recommended by NIST to replace DES. It was selected in 1997 after a competition to select the best encryption standard. .It's the hardest symmetric algorithm to crack, i.e providing the highest security, with a good time processing, holding the second faster algorithm after Blowfish algorithm.

The Origin of AES

US NIST issued call for ciphers in 1997:

15 candidates accepted in Jun 98

AES was selected in Oct-2000 as the winner and thus the new strongest standard.

Where Requirements for the new standard were:

128-bit data, 128/192/256-bit keys

Stronger and faster than 3DES

Active life of 20-30 years (+ archival use)

Both C and Java implementations

After testing and evaluation NIST have released all submissions & unclassified analyses, shortlist top 5:

MARS (IBM) -complex, fast, high security margin.

RC6 (USA) -v. simple, v. fast, low security margin.

Rijndael (Belgium) -clean, fast, good security margin.

Serpent (Euro) -slow, clean, v. high security margin.

Twofish (USA) -complex, v. fast, high security margin.

Rijndael(AES)obtained 86 votes, Serpent obtained 59 votes, Twofish obtained 31 votes, RC6 obtained 23 votes, and MARS obtained 13 votes.

Conclusion

According to putting the mentioned symmetric encryption algorithms to a trade-off analysis of difficulty to attacks , speed, performance , It was found that AES algorithms with their standard key lengths 128, 192, 256 are the most difficult to crack algorithms achieving the highest security, with a high speed .
And that is why no wonder, it is the most encryption algorithm commonly used to encrypt the contents of a message.
Thus, AES-128 was our choice to have the highest secure symmetric algorithm making no sacrifice of high speed execution.
high speed execution.

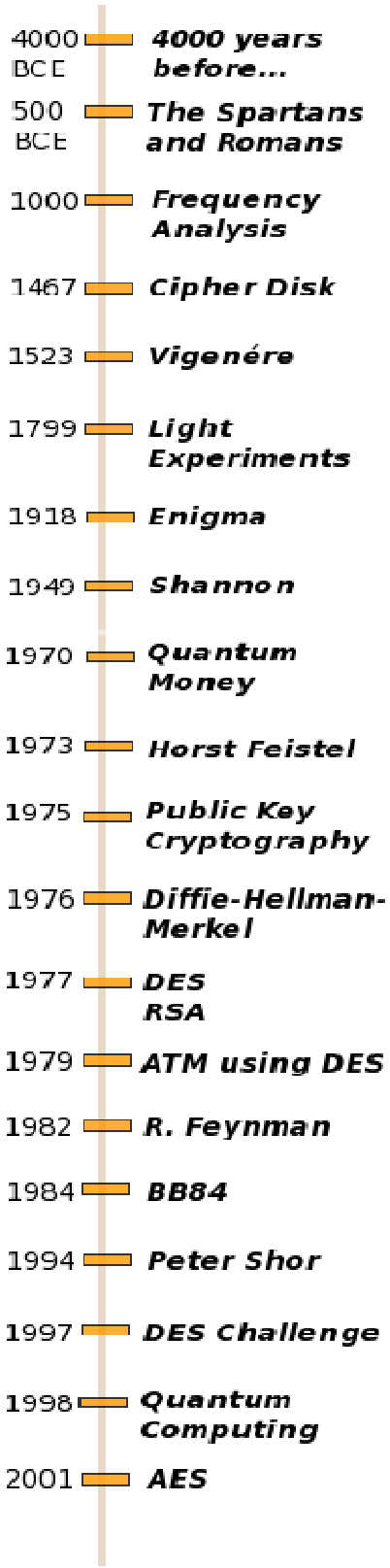


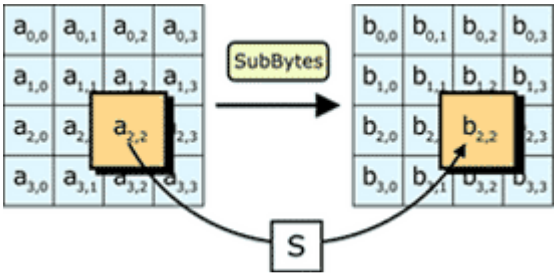
Fig 2.18

The figure represents the evolution of encryption algorithms.

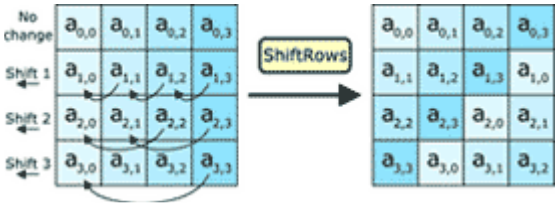
AES

Advanced Encryption Standard is a FIPS (Federal Information Processing Standards) algorithm and signifies a standardized encryption algorithm. AES may be used by the United States Government departments and agencies in order to encrypt information up to the Top Secret Level. Typically the title of AES is contested among the widest array of competing algorithms. Its basic idea is handling plain data as 2D bytes arrays elements performing shifting, transposing, multiplying them with integers to scramble any trace of the real data.

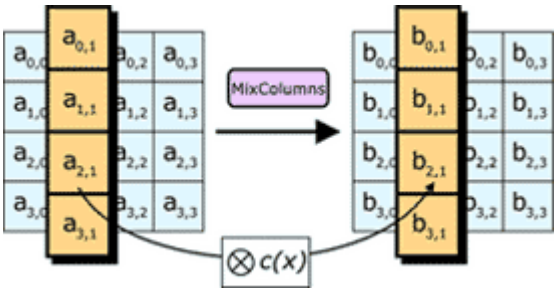
Graphical representation of how AES four steps work:



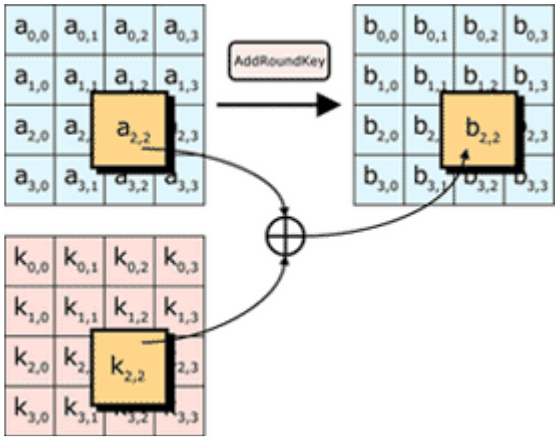
Step 1: SubBytes
Each byte in the state is replaced with its entry in a fixed 8-bit lookup table, S; bij = S(aij).



Step 2: ShiftRows
Bytes in each row of the state are shifted cyclically to the left. The number of places each byte is shifted differs for each row.



Step 3: MixColumns
Each column of the state is multiplied with a fixed polynomial c(x).



Step 4: AddRoundKey
Each byte of the state is combined with a byte of the round subkey using XOR operator .

Chapter Three
Analysis

The analysis phase has been of great value to our project.
Our system analyst collected all the data needed for the developers to start designing the program and drawing the route to be followed by the group in order to create the perfect solution.

After long research including field trips and a few meetings with prospective users of our solution, our system analyst came up with important results.

Here we discuss his most important conclusions.

Problem analysis

Documents are vulnerable to many ways of manipulation.

For example, we need to be sure that a check being paid in a bank has the original due value written by the entity or person who wrote the check.
We need to confirm the delivery of a message sent by company A to company B to send them the new product they are about to introduce before the competitors know about it.
There are many cases when we do not want anyone to know about the deals we are making, or secrets we share with others.

Current solutions

Today you cannot trust any person or entity while signing a deal with them. They could forge your signature, or manipulate the contract after you have already signed it.

Stamps, fingerprints, signatures, seals, third parties etc. are all solutions to retrieve the lost trust between people in our world.
But these solutions lack efficiency and speed. It is very easy to fake a signature, very easy to steal a temp and very easy to pride a third party.

These were not the users' only fears as documents manipulation can be greater problems. Sometimes a person can be a victim of a fraud because his signature is faked on an important legal document.

Objectives

1- Securing the exchange of documents between dealing parties.
2- SPOTTING any change in the original data to avoid any manipulation from one of the involved parties or a third sabotaging party.
3- Guarantee that only authorized entities will receive the documents.
4- User friendliness.
5- Parties do not have to be in the same place or at the same time to finalize the deal.

Recommendations

The analyst made his recommendations for the tools to use during the developing processes.
His recommendations came as following:

- Java language is best for building the program.
- For database we use SQL.
- Use the RSA, DES or AES encryption algorithms.
- HTML for creating documents forms.
- 2D barcodes.

Alternative solutions

The system analysis process made us aware of a few different solutions that can help some of the users with their problems.

We were able to have a full idea about users' needs, thanks to the interviews and meetings made by our analyst.

Some users need more simple solutions to limit the costs or requirements even if this means less efficiency or wastes more time.

Note

A lot of solutions were suggested but since that not all the users our analyst met were from technical backgrounds, most of these suggestions were not reasonable or technically impossible.

So we had to reduce them to our most recommended ones.

These are our most recommended alternatives

1- Do nothing.
   People just keep using handwritten documents and signatures, using temps and writing documents in the presence of all the concerned parties.

   Advantages:
   Cheap, no computers needed and closed minded users find more relief signing the document in the presence of the other party.

   Disadvantages:
   Slow, since it needs both parties to be present in the same time.
   Vulnerable to all kinds of forgery and manipulation.

2- Computer enabling:
   Inserting the computer role in the middle between the two parties.

   Advantages:
   Fast, since the presence of the two parties is unnecessary.
   Simple and any user can use it.

   Disadvantages:
   Manipulation of the data in the document is possible and this makes identifying the original document impossible.
   This solution offers no levels of security.

3- Implementing a barcode.
   Creating a barcode that represents the data in the document.

   Advantages:
   Useful if the documents used are not confidential.

   Disadvantages:
   If the user needs confidentiality of the document then this solution is useless because anyone who has a barcode reader can read the document.

Expected outputs of the program

1- An empty document form for the user to fill.
2- A barcode.
3- The final signed document with the barcode embedded.

Expected inputs of the program

1- User secret information from his USB storage.
2- Data filled by the user in the HTML document.
3- The barcode image (in case of decoding).

Archiving policies

The user is advised to backup the database and the HTML forms every ten to fourteen days.
This helps in keeping the data safe in case of disasters.

Problem Statement:

The following conclusions were reached by our analyst either from prospected user meetings, or from his own experience.

Although the solution is perfect for companies, banks and governments, it is not a solution for people who do not use computers.

Requirements

Hardware requirements

- A personal computer.
- High resolution printer (Optional).
- High resolution scanner (Optional).

Software requirements

- All operating systems (Windows, Mac, Linux, UNIX…etc).
- Java Runtime.

Chapter Four
Results and Discussion

Studying the concerns and issues experienced by people and entities who use documents every day, many defects and insufficiencies in the legal documents trip between creation, signing and archiving can be found.

Many of these concerns can be resolved by using the solution discussed in this documentation.

The project's database

Consists of two separate databases:

- Administrators database
- End users database

1-Administrator database

Contains: users table, forms table

Users table:

Users-ID	Users-Signature	users_firstname	users_lastname	users_publickey

users_modulus	users_privatkey	telephone number	address	fingerprint

Forms information table:

Form-ID	form_name	form_link	form_view path	data_path

Approvement_form_path

2-End user database

End_User_table

Contains some end user related fields from the Administrator database Users table.

User-ID	User-signature	first_name	last_name	public_key	modulus

forms_information_table:

Is an exact Replica of the administrators contracts table.

Form-ID	form_name	form_link	form_view path	data_path

Approvement_form_path

The end user data base is updated frequently from the Administrator database to insure that the latest added users and forms are available for the end users at any time.

The description of data elements

1-Administrator database:

Data Types and Description		
Documents/Field Names	**Type**	**validations**
Users table:		
• Users_ID	Auto Number	The private key of the table is unique
• Users_signature	Text	Foreign key is unique
• User_firstname	Text	Presence validation
• users_lastname	Text	Presence validation
• users_publickey	Number	Presence validation
• users_modulus	Number	Unique- Presence validation
• users_privatkey	Number	Unique- Presence validation
• telephone number	Number	Unique- Presence validation
• address	Text	Presence validation
• fingerprint	Attachment	Presence validation

Data Types and Description		
Documents/Field Names	**Type**	**validations**
Forms information table:		
• Form_ID	Auto Number	The private key of the table is unique
• Form_name	Text	Presence validation is unique
• Form_link	Text	Presence validation is unique
• Form_view path	Text	Presence validation is unique
• Data_path	Text	Presence validation is unique
• Approvement_form_path	Text	Presence validation is unique

2) End user database:

Data Types and Description		
Documents/Field Names	**Type**	**Validations**
End_Users_table:		
• Users_ID	Auto Number	The private key of the table is unique
• Users_signature	Text	Foreign key is unique
• User_firstname	Text	Presence validation
• users_lastname	Text	Presence validation
• users_publickey	Number	Presence validation
• users_modulus	Number	Unique- Presence validation

Data Types and Description		
Documents/Field Names	**Type**	**Validations**
Forms information table:		
• Form_ID	Auto Number	The private key of the table is unique
• Form_name	Text	Presence validation is unique
• Form_link	Text	Presence validation is unique
• Form_view path	Text	Presence validation is unique
• Data_path	Text	Presence validation is unique
• Approvement_form_path	Text	Presence validation is unique

Connecting the data base to JAVA

The connection is created using **ODBC** which stands for Open Data Base Connectivity.

ODBS

It is a connection that is created to define a connection between a computer and a database stored on another system. The ODBC connection contains information needed to allow a computer user to access the information stored in a database that is not local to that computer.

Creating an ODBC database connection

You need to define the type of the database application – like for Example Microsoft SQL or Oracle or Access or mySQL. Once you have defined the type of database you need to select or supply the appropriate driver for a connection (Windows already contains many of these) and then supply the name of the database file and the credentials needed to access the database. Once the ODBC connection is created, you can tell specific programs to use that ODBC connection to access information in that database.

It is difficult for the user to create an ODBC connection manually, so we created a batch file that automatically configures it.

Here are some java code examples on how java connects to execute a query.
Example 1:

A connection to the data base then creating a query to return all values of form names in a combo-box and ordered by form name (shown in fig 1)

```
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

Connection con=DriverManager.getConnection("jdbc:odbc:test","","");
Statement s=con.createStatement();
s.executeQuery("SELECT Contract_Name FROM Tbl_Contracts ORDER BY Contract_Name ");
ResultSet rset = s.getResultSet();



 while(rset.next())
     {
        String x = rset.getString("Contract_Name");
        System.out.println(x);
        jComboBox1.addItem(x);
       }
```

Fig 4.1

Example 2:

Connecting to execute a amethod displayQueryResults and previewing the results on a Jtable return values of user_name (his signature) Users_First_Name and Users_Last_Name (shown in fig 2)

```
public void keyPressed(KeyEvent e)
 {
 displayQueryResults("SELECT User_Name,First_Name,Last_Name FROM (SELECT User_Name,First_Name,Last_Name FROM Tbl_End_User WHERE User_Name LiKE '" + query.getText() + "%' ORDER BY User_Name) WHERE First_Name LiKE '" + query2.getText() + "%' AND Last_Name LiKE '" + query1.getText() + "%' ORDER BY First_Name" );
        }
```

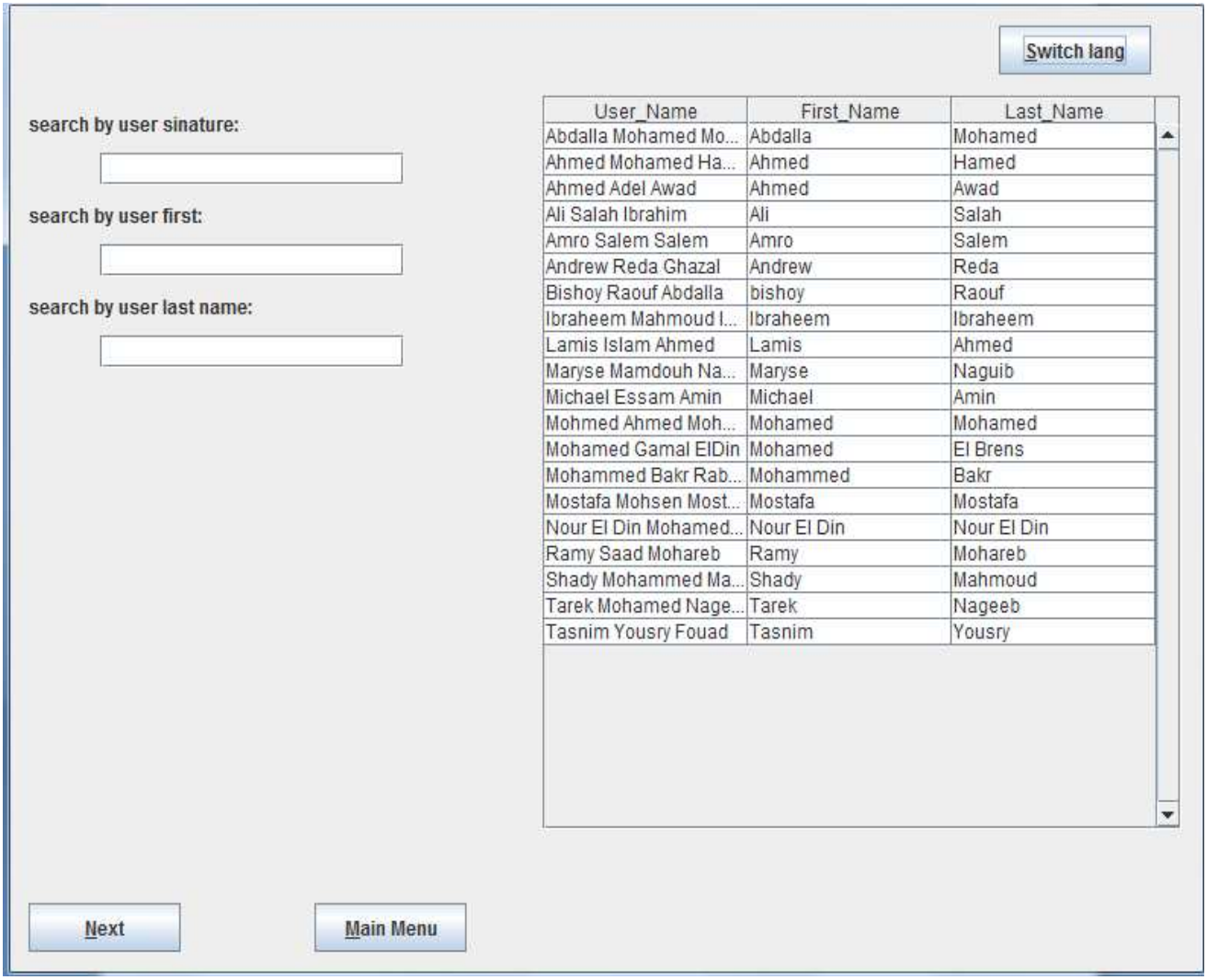


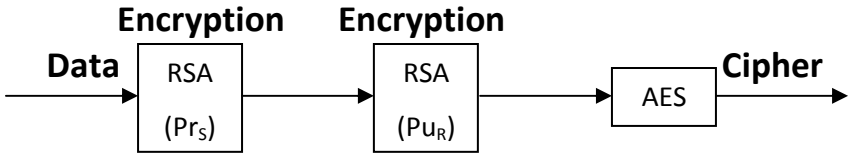
Fig 4.2

Cryptography

Our encryption goes through two Phases

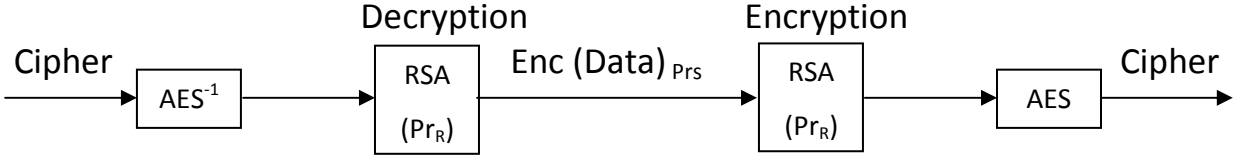
Phase I: For generating (waiting to be approved) barcode after the sender fills the form, user's data is encrypted as follows

1) RSA Encryption using the sender's private key.
2) RSA Encryption using the receiver's public key.
3) AES symmetric encryption.



Phase II: If the Receiver approves the form

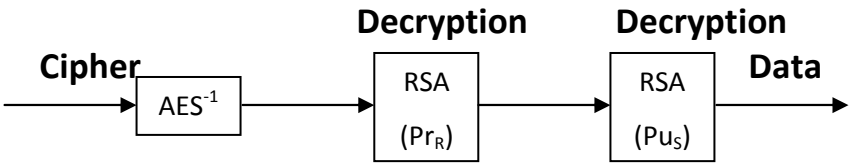
1) Decrypt the cipher decoded from the barcode using AES decryption stage.
2) Decrypt the output with RSA using the receiver's private key.
The usage of this stage is: the generated cipher now is only the filled data encrypted with the sender's private key, in other words the cipher now holds the approval of the sender.
3) Encrypt the output of the previous stage with RSA again using the receiver's private key. At this stage the receiver adds his/her approval to the approval of the first party, i.e. now the cipher does hold the approval of both parties together.
4) Final stage is encrypting the full approval of the two parties using AES.



The two decryption phases work as follows

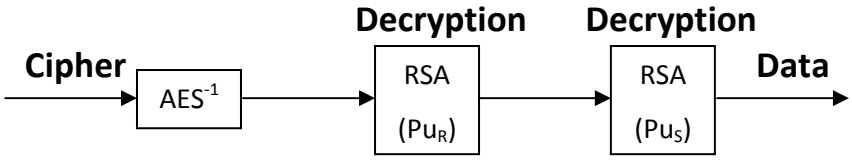
I) For an unapproved barcode (To show up the encrypted data in the form to be either approved by the receiver or declined):

1) Decrypt the decoded cipher from the barcode using the AES decryption stage.
2) Decrypt the output with RSA using the receiver's private key.
3) Final stage is RSA decryption with the sender's public key



II) For decrypting the data of an approved barcode (This stage can be made by a third party as well to check whether BOTH parties agreed and signed the form or not):

1) Decrypt the decoded cipher from the barcode using AES decryption.
2) Decrypt the output with RSA using the receiver's public key (Available for a third party).
3) Final stage is RSA decryption with the sender's public key (Also available for a third party).



Conclusion:

Both encryption and decryption stages seem to be complex. But the complexity in our application is a MUST due to many reasons:

- There should be NO WAY for any party, with using our software, to deny its awareness of every single detail about the form and the filled data in it.
- There should be NO WAY for any party to deny itself, or deny approving the form in any situation.
- Judging is available, i.e. the two parties know that in any case of prejudice ,not only a judging side can get the conspirator ,but anyone having the software and the disputed barcode ,can check if the documentation form is approved by the two parties TOGETHER or not .
- Additionally, no one can claim that the data is changed in the middle, because it is encrypted after the RSA with AES, so the data is emphasized more by an uncracked algorithm.

The Operating Software

A simple and user friendly interface that guides the user through the different phases of the processes.
It plays the role of the brain in our project.

The solution helps in securing documents against forgery and manipulation.

It offers faster and safer exchange of documents, full authentication between sender and receiver as the encryption technique, accessibility to authorized users only delivery confirmation, and many others tasks that meet the users' expectations and sometimes exceeding them.

These serious jobs need a powerful program that is to adequate liability as the tasks it does.

Security Using Digital Barcodes (SUDB) is the name we chose for the program that brilliantly operates the project's different processes in a smooth sequence helping the user through all steps needed to reach his aim easily and without any problems.

Logic issues

Creation of documents and signing them by the involved parties is a process that is divided into many steps. These steps must take a specific sequence in order to be legal.

This sequence was studied carefully before developing, in order to avoid logical problems and to help the user perfectly.

The program is written in Java programming language.
Java was chosen over C language because of its portability.

Java has many advantages; we mention some of them here

- Java is platform-independent, and can move easily from one computer system to another.
- Java is simple: Java was designed to be easy to use and is therefore easy to write, compile, and debug than other programming languages.
- Java is object-oriented: Allowing us to create modular programs and reusable code.
- Java is distributed: Distributed computing involves several computers on a network working together.
- Java programs are compiled into Java Virtual Machine code called bytecode. The bytecode is machine independent and is able to run on any machine that has a Java interpreter.
- Java is secure: Java is one of the first programming languages to consider security as part of its design.
- Java is robust: Robust means reliable and no programming language can really assure reliability.
- Java is multithreaded: Multithreaded is the capability for a program to perform several tasks simultaneously within a program.

The result is a light, fast and portable program. Written by the members of our group and reviewed by many prospected users who gave us some tips and helped us enhance our design many times through the development process.
Please see the user involvement forms in the appendix section.

Each member of the group was given a certain task, then all the pieces were integrated together to bring the program to life in its final form.

The following pages are about the program in details.
Its design, tasks and sequence are all discussed in details giving you a complete and clear idea about the program and its functions. The discussion is presented with some snapshots taken from the original interface to help understanding the program very well.

Requirements

The program will be operated by both, technical and nontechnical users including bank tellers, salesmen, individuals etc.

So, an interface that all the users can run without having to ask for technical help is needed. This was the most important requirement.
Some other requirements are also needed, here are some of them

- Robustness
- Reliability
- Speed
- Accurate outputs
- Least possible inputs

All the mentioned requirements were elegantly met in the program, creating a user friendly atmosphere without affecting the program's efficiency or speed.

The program also went through many phases of testing and user consultancy to guarantee its suitability to different users with different educational backgrounds.

Interface language

In order to meet the users' none ending expectations, our developer delivered the program interface in two different languages.

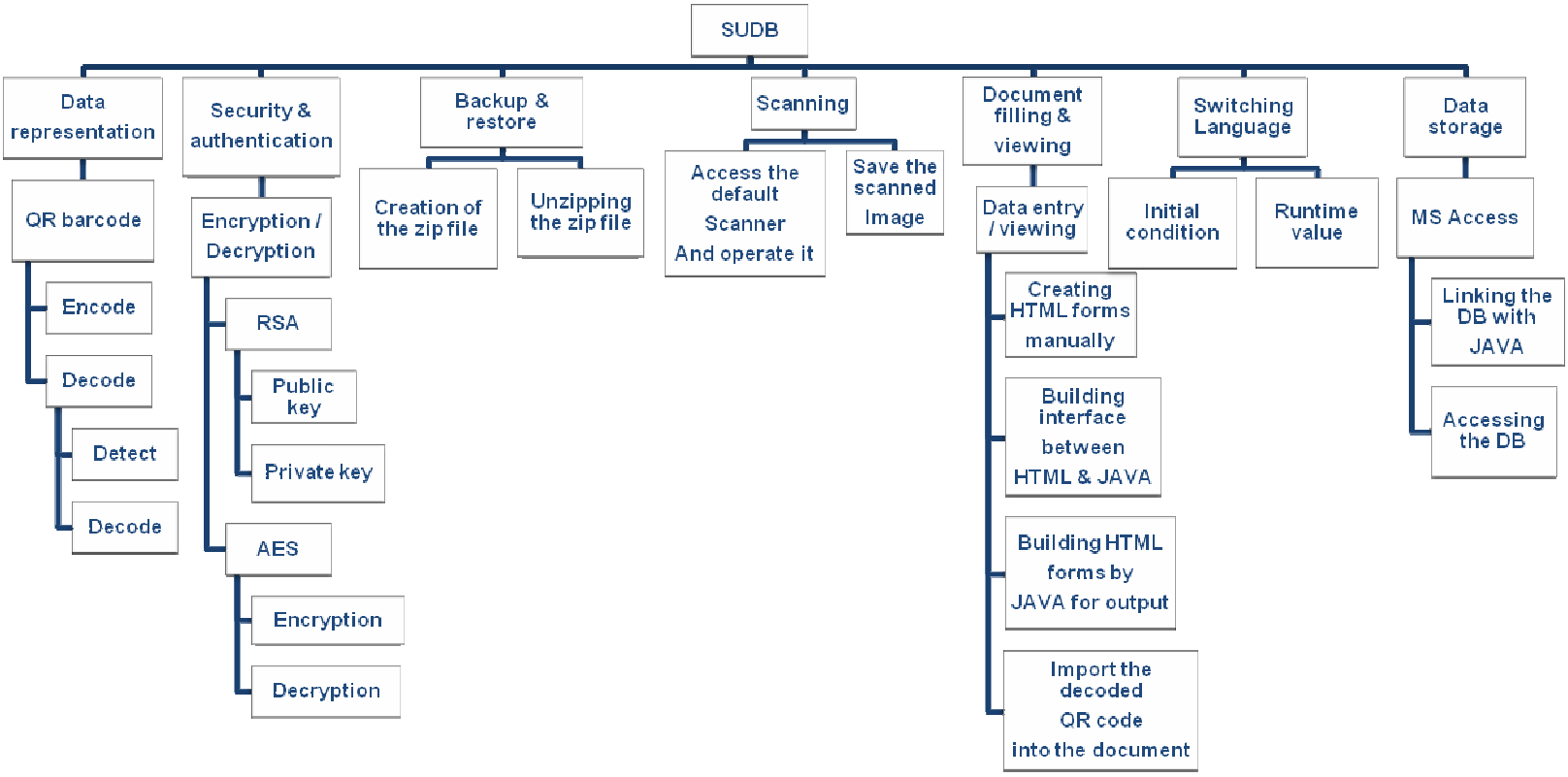
1- English language.
2- Italian language.

This step proved great success in achieving users' convenience according to the user involvement forms.

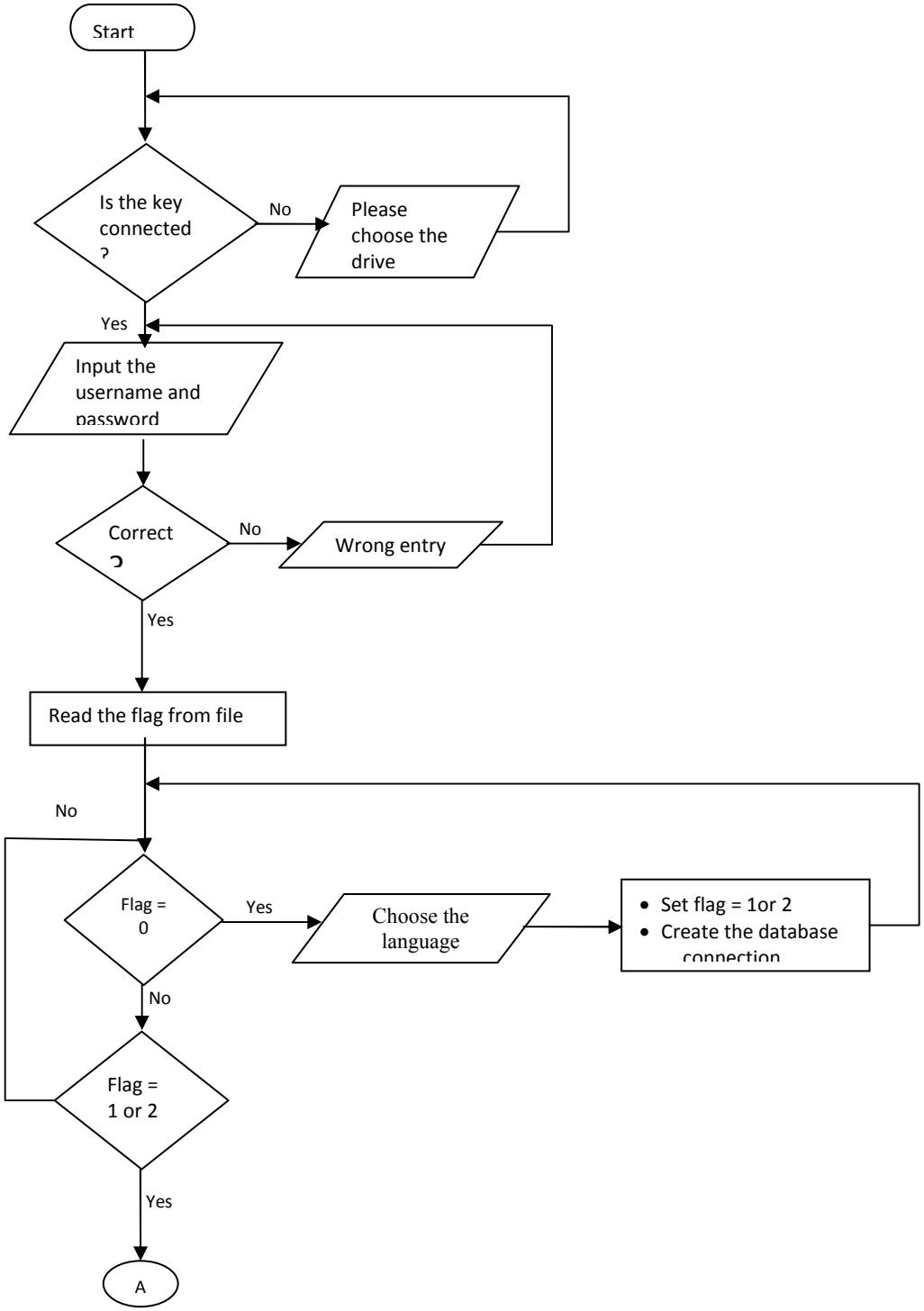
Design

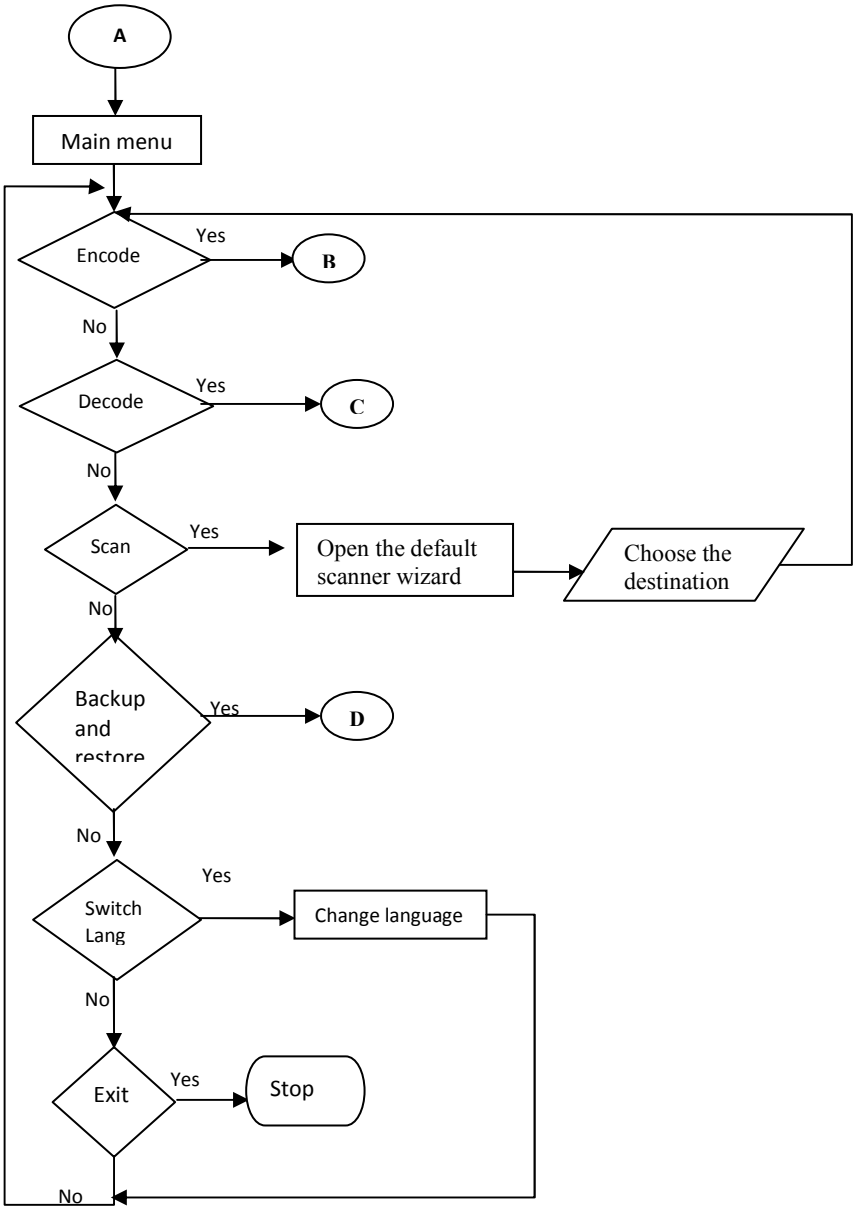
This is a brief explanation of what this program does, and this same topic will be covered in details soon.

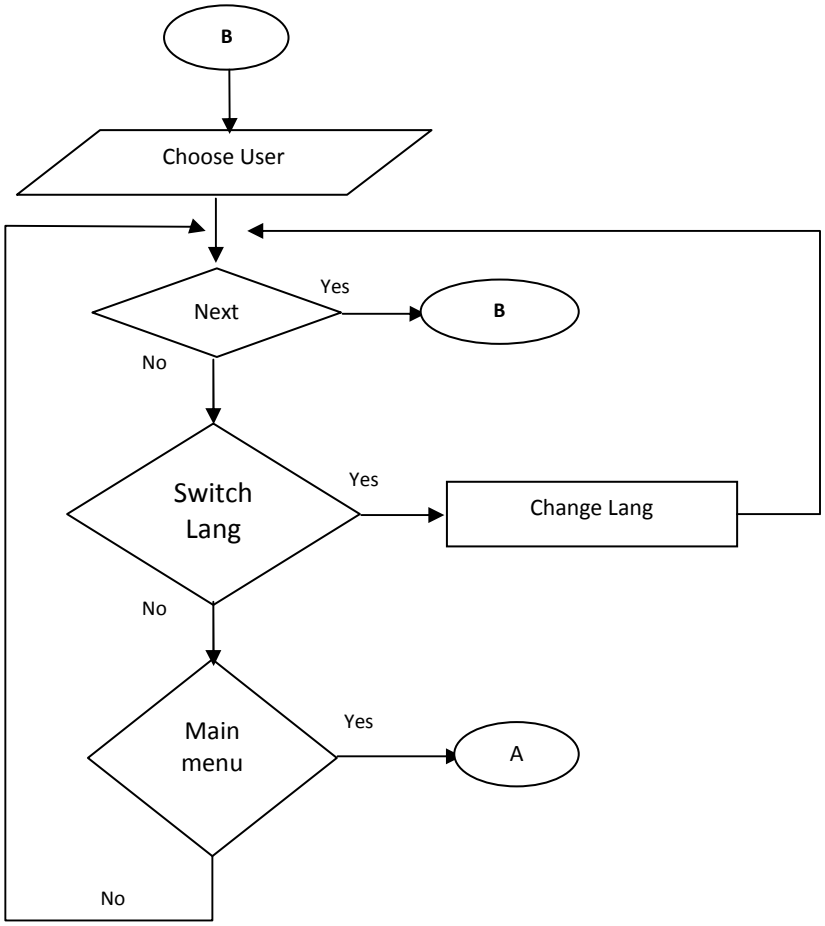
Jackson diagram

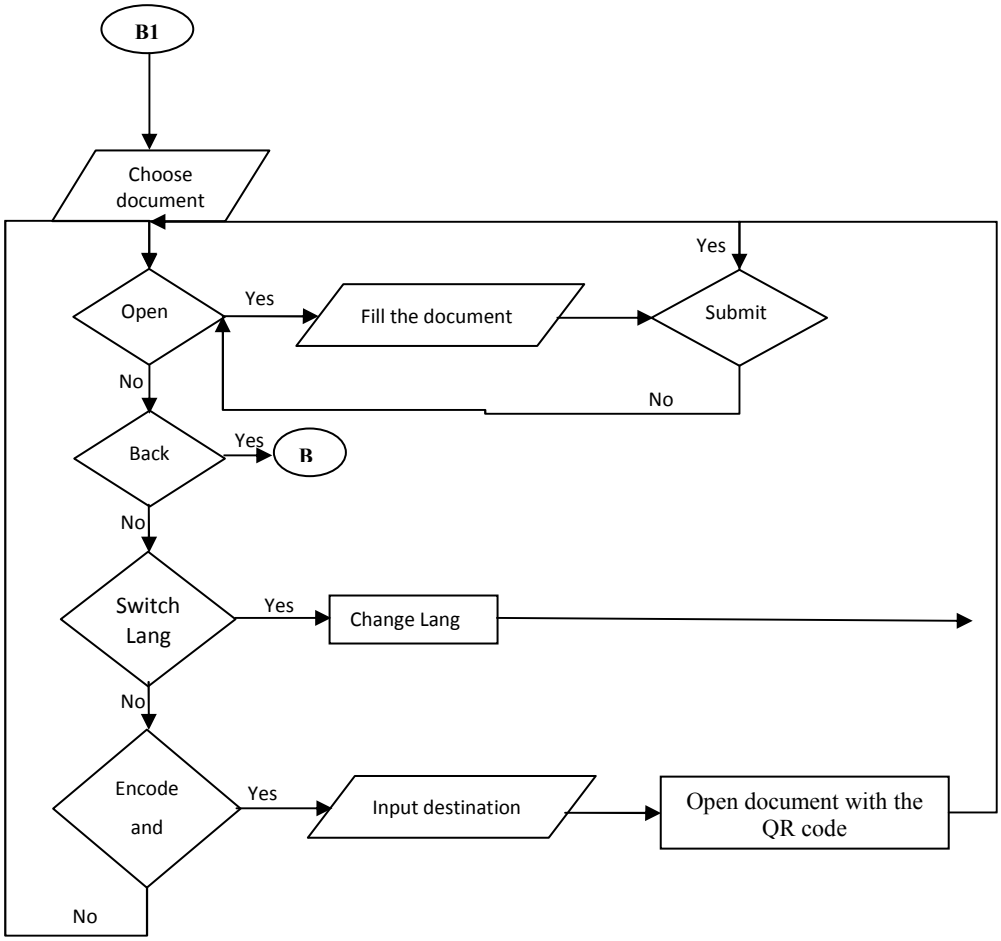


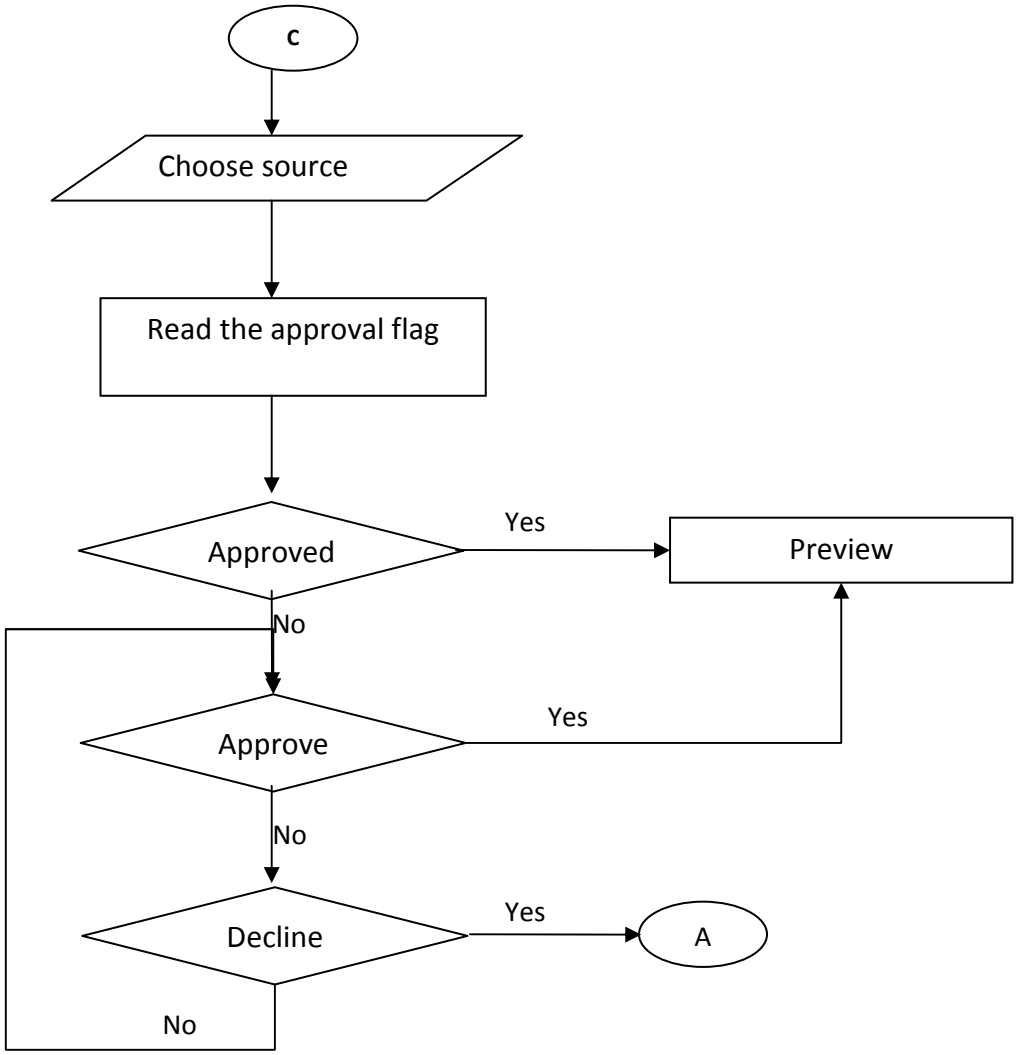
Program flowcharts

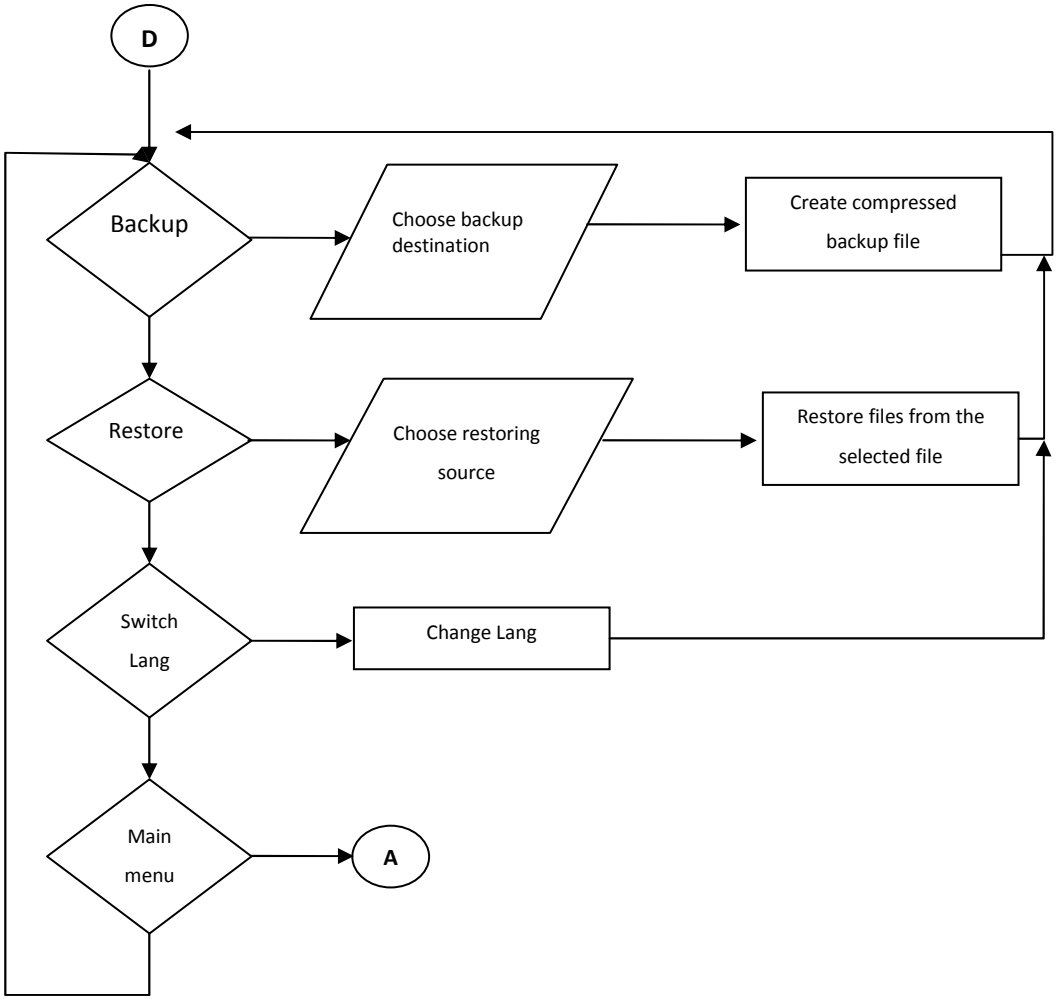



A
Main menu
Encode
Yes
B
No
Decode
Yes
C
No
Scan
Yes
Open the default scanner wizard
Choose the destination
No
Backup and restore
Yes
D
No
Switch Lang
Yes
Change language
No
Exit
Yes
Stop
No

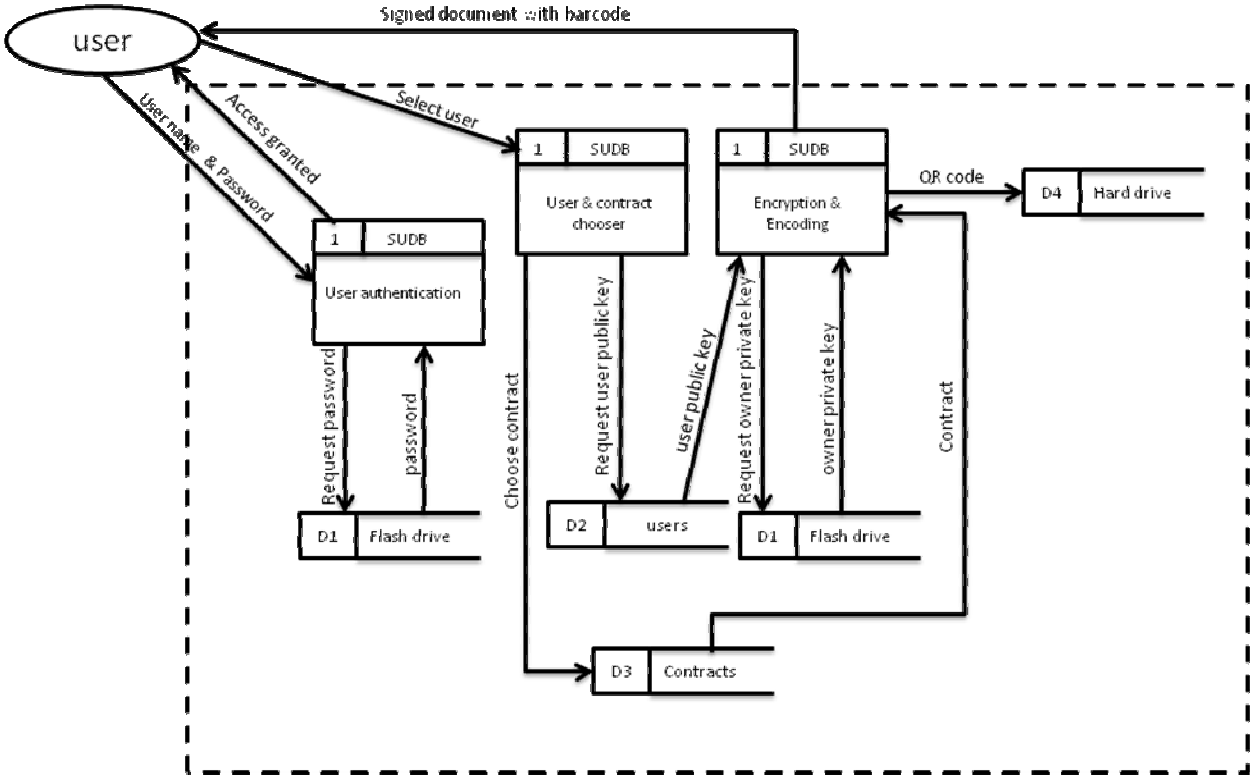

B
Choose User
Next
Yes
B
No
Switch Lang
Yes
Change Lang
No
Main menu
Yes
A
No

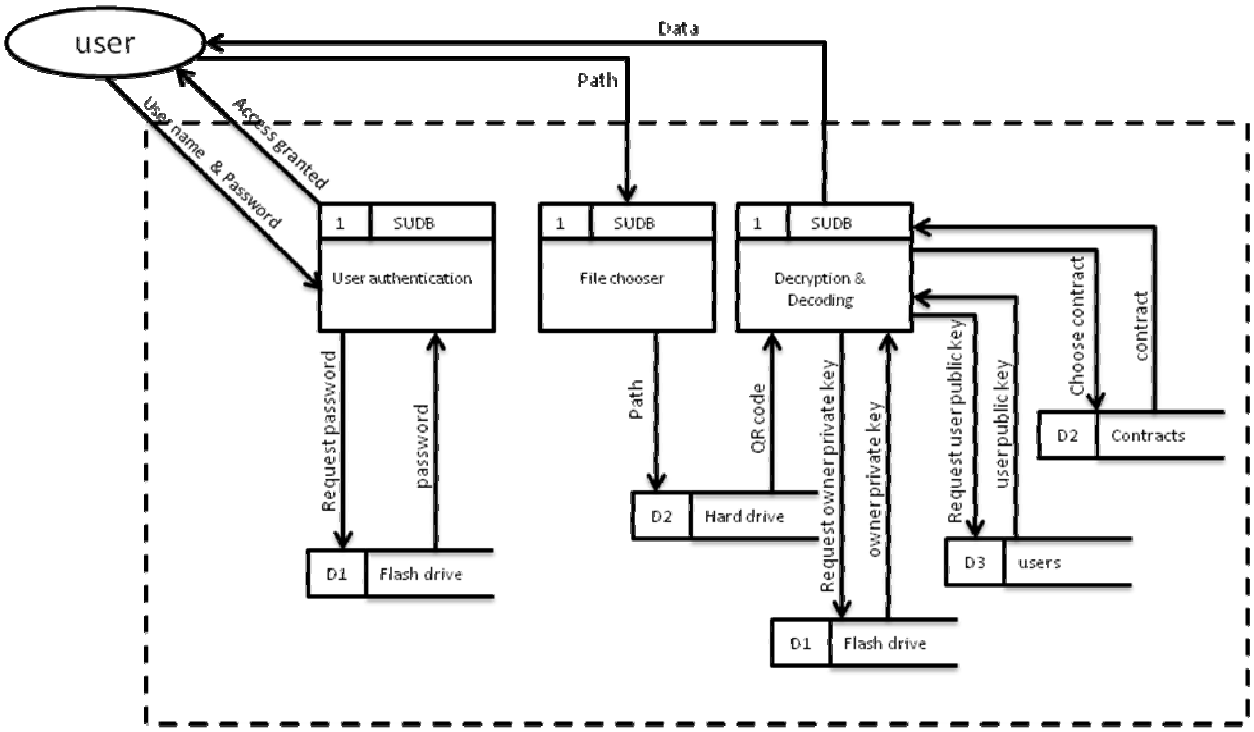

B1
Choose document
Open
Yes
Fill the document
Submit
Yes
No
No
Back
Yes
B
No
Switch Lang
Yes
Change Lang
No
Encode and
Yes
Input destination
Open document with the QR code
No


C
Choose source
Read the approval flag
Approved
Yes
Preview
No
Approve
Yes
No
Decline
Yes
A
No


D
Backup
Choose backup destination
Create compressed backup file
Restore
Choose restoring source
Restore files from the selected file
Switch Lang
Change Lang
Main menu
A

Data flow diagrams





Starting the program

The first time the program runs, it asks the user to choose the default language. Although the language choosing window never shows again, the user can still change the language anytime inside the program, just by clicking the button assigned for this job.

The program then starts by asking the user to enter his username and password (fig 4.3) then to insert the personal USB storage device if it is not already inserted (fig 4.4). This USB device is given to the user by our company (still virtual). It saves his ID, username, password, public and private keys and the modulus used in the RSA encryption discussed pre.

The program then compares the username and password entered by the user with the ones on the USB storage device.

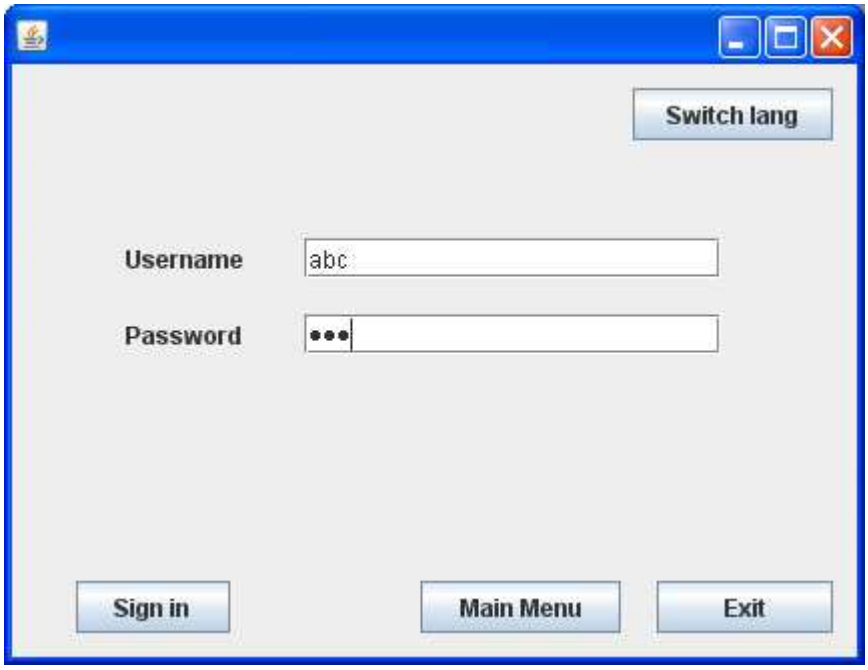


Fig 4.3

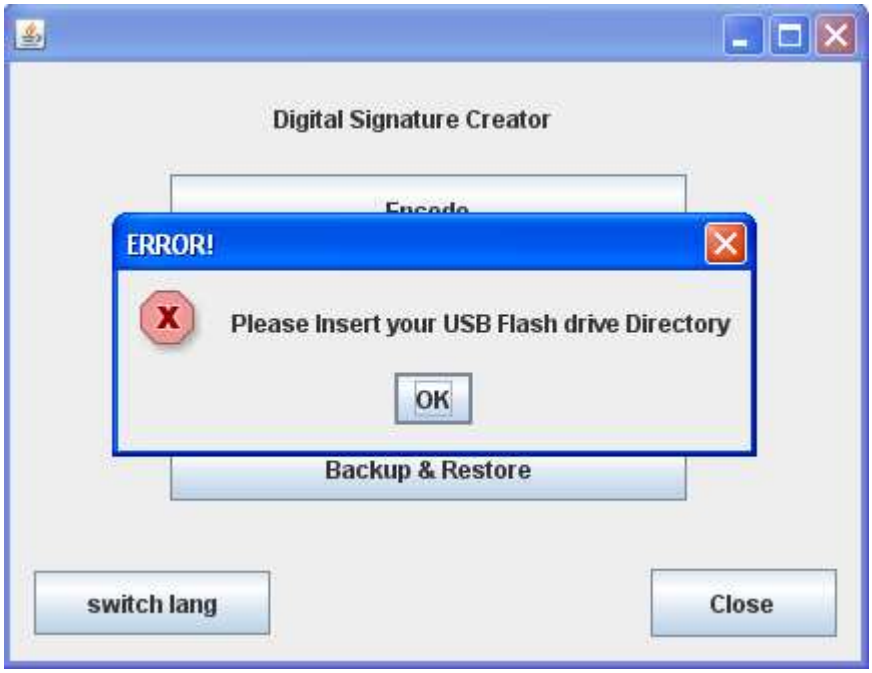


Fig 4.4

If the username and password entered by the user are identical to those saved in the USB device, then the user is authenticated and granted access to the main menu window (fig 4.5) which is considered to be the gate to all processes held by the program.

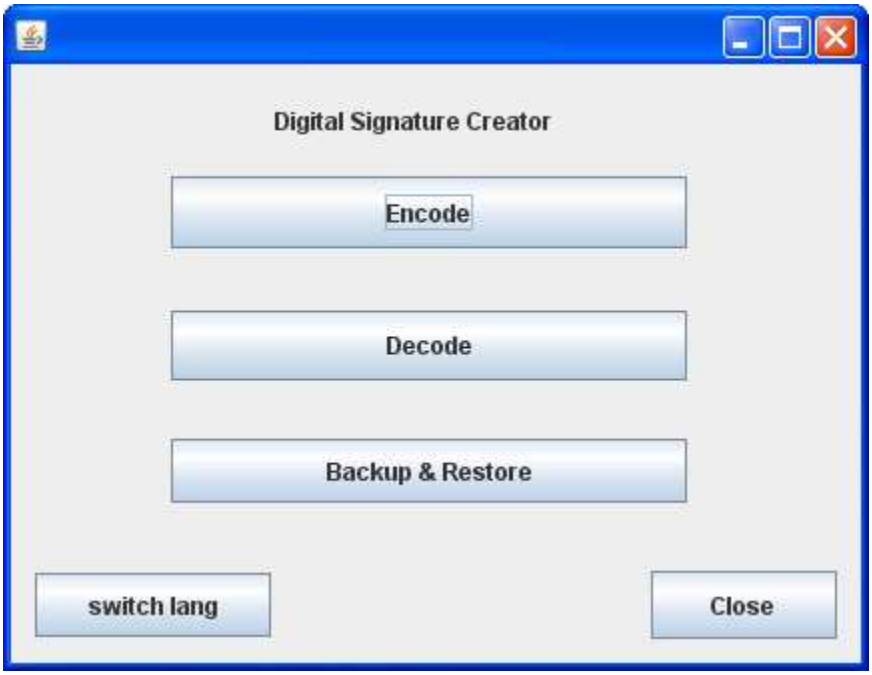


Fig 4.5

From the main menu window, the user can start the Encoding, Decoding, and Backup processes.

The Encoding Process

If the user clicks “Encode”, the encoding process is started.
The encoding process includes encryption processes.

A database is opened (fig 4.6) for the user to choose the other party he is going to make the deal with.
Searching the users is possible (username, first name and last name are the possible search fields).

“Next” button takes the user to the next stage (choosing a form to encode).

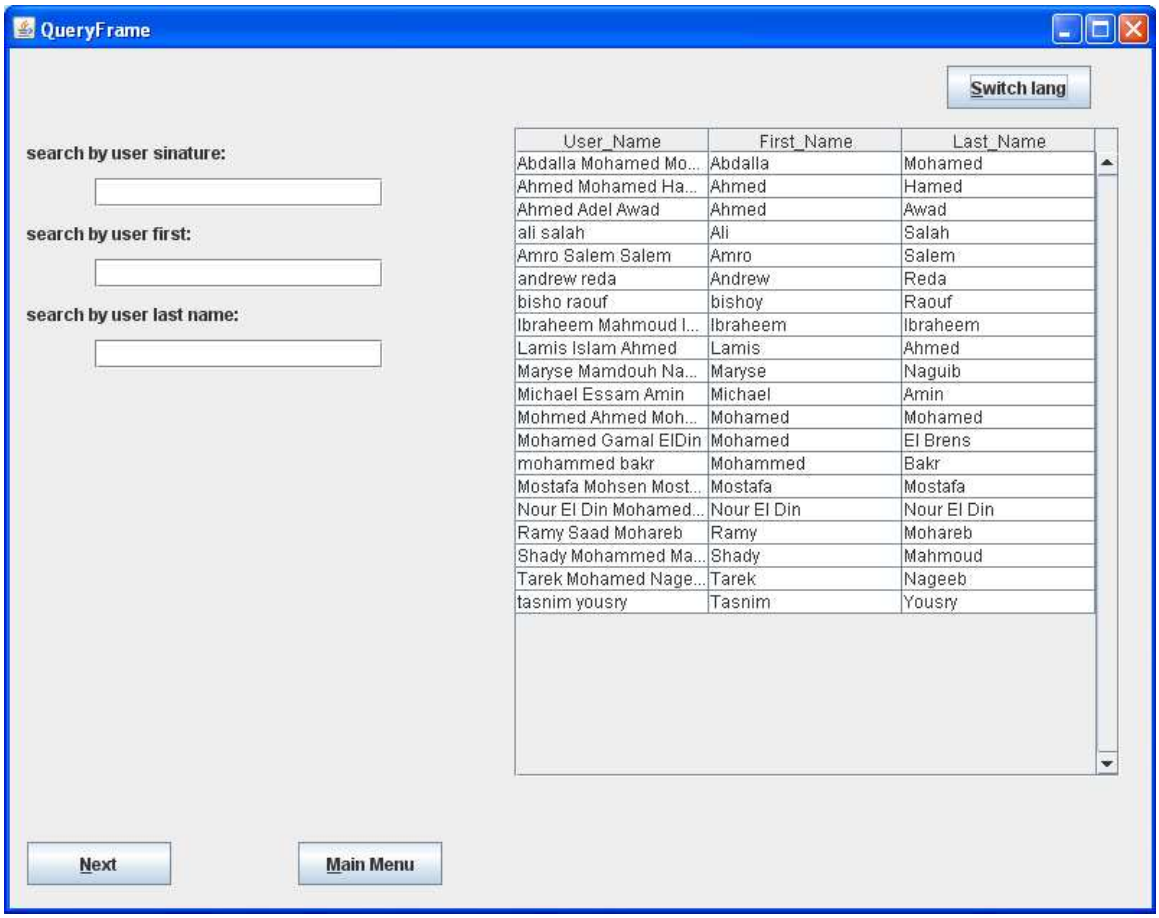


Fig 4.6

The form selection window (fig 4.7) appears giving the user the choice between the available forms.
The chosen form is open in an HTML page for the user to fill, submit and save.

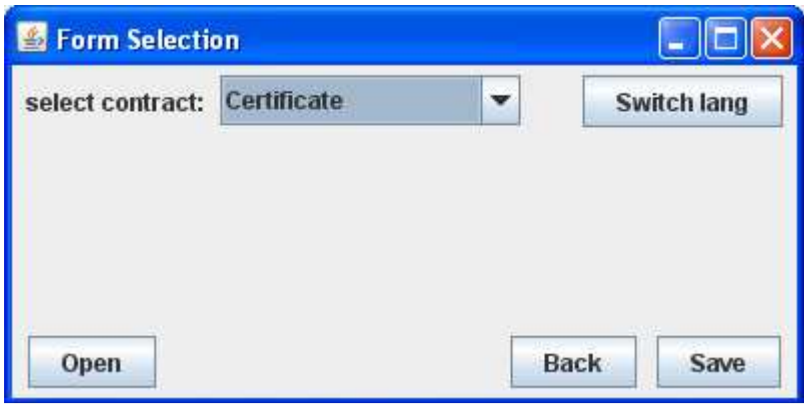


Fig 4.7

Encryption

After clicking “save”, the data filled in the form is converted into ASCII code.

The first encryption phase is then implied (to generate the primitive barcode to be accepted or denied by the other party).

Recalling it in a nutshell:

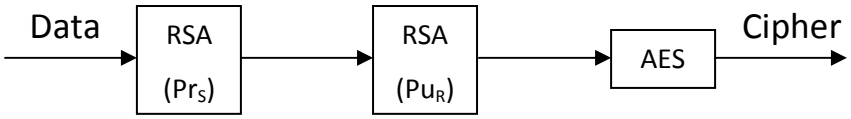


Fig 4.8

After that the barcode is created using the encoding algorithm.

The result is a unique and immune barcode that contains the data filled in the document. This barcode can be decoded *AND* decrypted only by the receiver.

The Decoding Process

Back to the main menu, if the user clicks “Decode”, the program starts the decoding process.

In this phase the decryption process is held.

A file chooser is opened asking the user to browse the barcode to be decoded.

The barcode is simply decoded using the decryption algorithm, but the output is not useful data (cipher). This cipher is then decrypted to the original data.

Decryption

There are two phases of decryption, there is a flag embedded in the barcode which lets the software know according to which phase this barcode should be decrypted, in other words decryption of approved barcodes is different from the waiting to be approved or declined ones.

Recall the 2 decryption phases:

For an unapproved barcode:

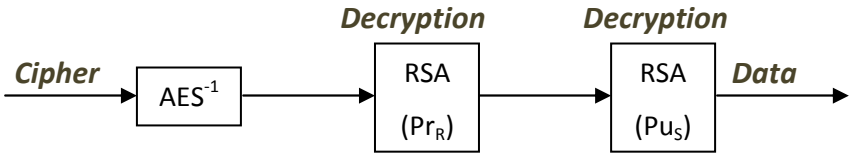


Fig 4.9

For an approved barcode, decryption works like this:

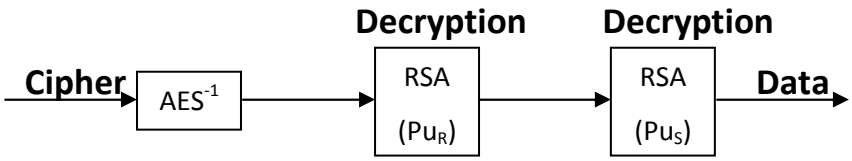


Fig 4.10

ASCII code is retrieved and then converted to the original data, then restored to the HTML form.

The HTML form is then filled and shown to the user to approve or deny.

In case that the user approves the second phase of the encryption is implied as follows:

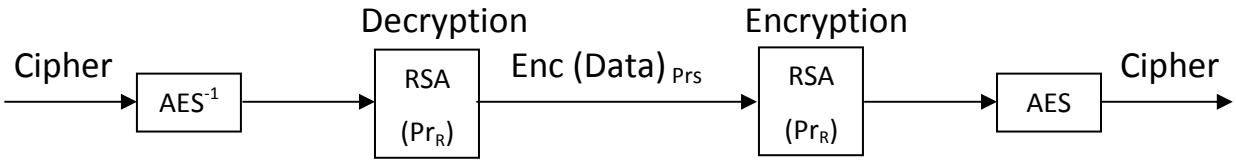


Fig 4.11

Note

The interface was reviewed by some prospected users (see the appendix) and their opinions helped in the developing process.

Software testing

Introduction

Software testing is as old as the hills in the history of digital computers. The testing of software is an important means of assessing the software to determine its quality. Since testing typically consumes 40~50% of development efforts, and consumes more effort for systems that require higher levels of reliability, it is a significant part of the software engineering.

Despite advances in formal methods and verification techniques, a system still needs to be tested before it is used. Testing remains the truly effective means to assure the quality of a software system of non-trivial complexity, as well as one of the most intricate and least understood areas in software engineering.

2.2 The Testing Spectrum

Testing is involved in every stage of software life cycle, but the testing done at each level of software development is different in nature and has different objectives.

First Unit Testing is done at the lowest level. It tests the basic unit of software, which is the smallest testable piece of software, and is often called "unit", "module", or "component" interchangeably.

1) QR-Barcode:

Encoding-Decoding: (1st library package)

When entering small amount of data there are no problems in the process

Fig 4.12

Barcode which contains text:
"This is a test" and outputs the same.

When entering ≈ 180 characters Decoder sometimes fails to retrieve data

Fig 4.13
A barcode contains ≈ 180 characters

```
run:
Decoding started
Drawing matrix.
Scanning Finder Pattern.
Not found, now retrying...
jp.sourceforge.qrcode.exception.DecodingFailedException: Invalid number of Finder Pattern detected
```

Fig 4.14
(A preview of the error message)

Decoder fails to retrieve data in several different cases. This was solved by using a different QRcode library and creating more advanced methods to get the saved data.

Encoding-Decoding: (2nd library package)

No problems in retrieving data of large amounts

Fig 4.15
(Barcode which contains text: “This is a test” and outputs the same)

Fig 4.16

A barcode contains ≈ 180 characters

This is a test, This is a test, This is a test,..

```
this is a test,this is a test,this is a test,this is a test,this is a test,this is a test,
this is a test,this is a test,this is a test,this is a test,this is a test,this is a test,
BUILD SUCCESSFUL (total time: 6 seconds)
```

Fig 4.17

Output of the QRcode

2) Security Algorithms:

Encryption-decryption: (Symmetric and Asymmetric Algorithms)

AES:

IN the AES Symmetric algorithm no problem was found in ciphering and deciphering.

RSA:

Encrypting once using a public or a private key was successful since the message is always smaller than the modulus shown in (Fig.), the problem was that we needed to split the message before the 2nd encryption with RSA to be sure that the input is always smaller than the modulus of the algorithm.

```
run:
the message is:          777972657777696832666575823277797265777769683282656669697296495296495096504 8
the encrypted message is: 136799420117484281402722701186508859272125731675098373362267393558558043302 5
the decrypted message is: 777972657777696832666575823277797265777769683282656669697296495296495096504 8
BUILD SUCCESSFUL (total time: 1 second)
```

Fig 4.18

Preview of input output of RSA

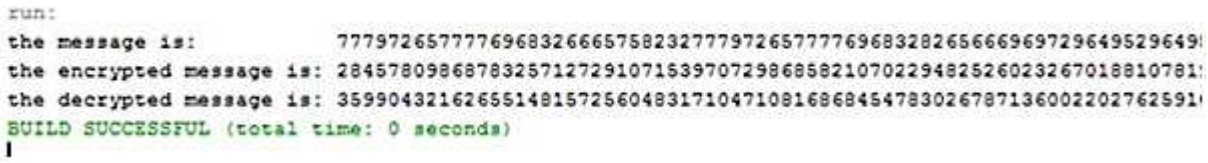


Fig 4.19

Fail to return the same value

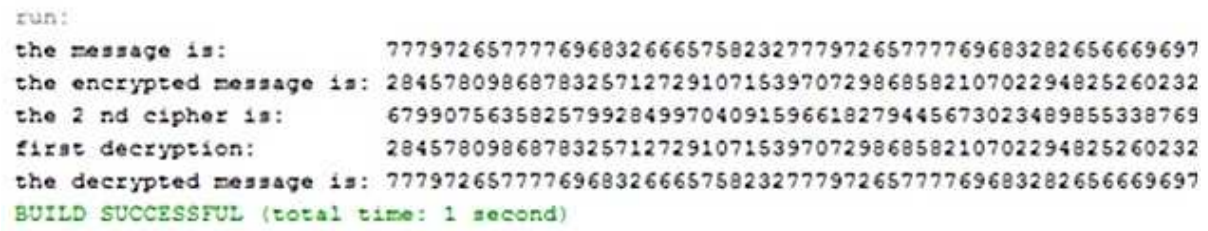


Fig 4.20

The output after splitting the data is the same

Integration Testing is performed when two or more tested units are combined into a larger structure.

The test is often done on both the interfaces between the components and the larger structure being constructed, if its quality property cannot be assessed from its components.

Encoding-Decoding of an encrypted message

The AES outputs Special characters which can't be retrieved by the decoder

The cipher is:

»wÖ??e¥fgJ)0?üþÿô)º¡ß¯:voØÛCk?z:BèdP³ÒR,ò!??ñÞÉªéÂ<I???oém?qa^??ùln¡Ü ?ÑÒr¨¯??º½Fµç:MÀÀ?-µÁM{?¹?/9A Yª¦ÎFr??¡p?UZû|Ñ¢6-ô|Î?>å$uB§tzu|ñ^}÷Ç µàù/ET¾?U}èÐ??Ù>q<ÁF¹Y-ËIK²?U»Wùê^<]Ccoÿíc%ßo@?§";

Fig 4.21
(Encoded cipher that fails to return same value)

The decoder returns the value of

*□w□??e□fgJ)0?□□□□)□□□:vo□□Ck?z:B□dP□□R,□!??□□□□□<I
???o□ma^??□In□□?□□r□□??□□□F□□:M□□?-□□M{?□?/9A□Y□
□□Fr?□p?UZ□|Ƅ6-□|□?>□$uB□tzu|□^}*□□□□/ET□?U}□□??□>q<
□F□Y-□IK□?U□W□□^<]Cco□□c%□o@?□

The AES have been changed to give only alphanumerical characters
The cipher is

wGSOguiFgTc2NxBCvI+Mx7jtD7wxQwfxQ8WFNH01pGCJxqYmyUOTkU8CtzRZ8Bak
ErsdXwqxXo9kpMY3i/Rnm7KP/BE20o5/+ehQxkb0N8nKHeS9/tuojITXehmIF/pRuiS
dC8z/1tfajn8DQxuust5btLp0s+Wc7hBpWIJvtsNtBnoYw7RggDleaJeD8xcuEchujw3a
oJdRwu7v9JfYTCU2XRRFaLe5qLA/+Zp7UAwXTFoASVSQKVZHdvRtP4DZq2J+i0pujkk
mhZq9mZHngxLtq+SEIa11LgBySZUOoyE=0

Fig 4.22(The Encoded cipher successfully returns)

Encoding-Decoding: (2nd library package)

No problems in retrieving encrypted data of large amounts but sometimes encoder creates barcodes with hidden patterns so the decoder fails to return value

Solved by creating recursive encoding (check if it can decode and re-encode a padded if it fails)
Re-encode with padding changes the shape of the QR-barcode so the decoder can locate and find the patterns required to retrieve the text.

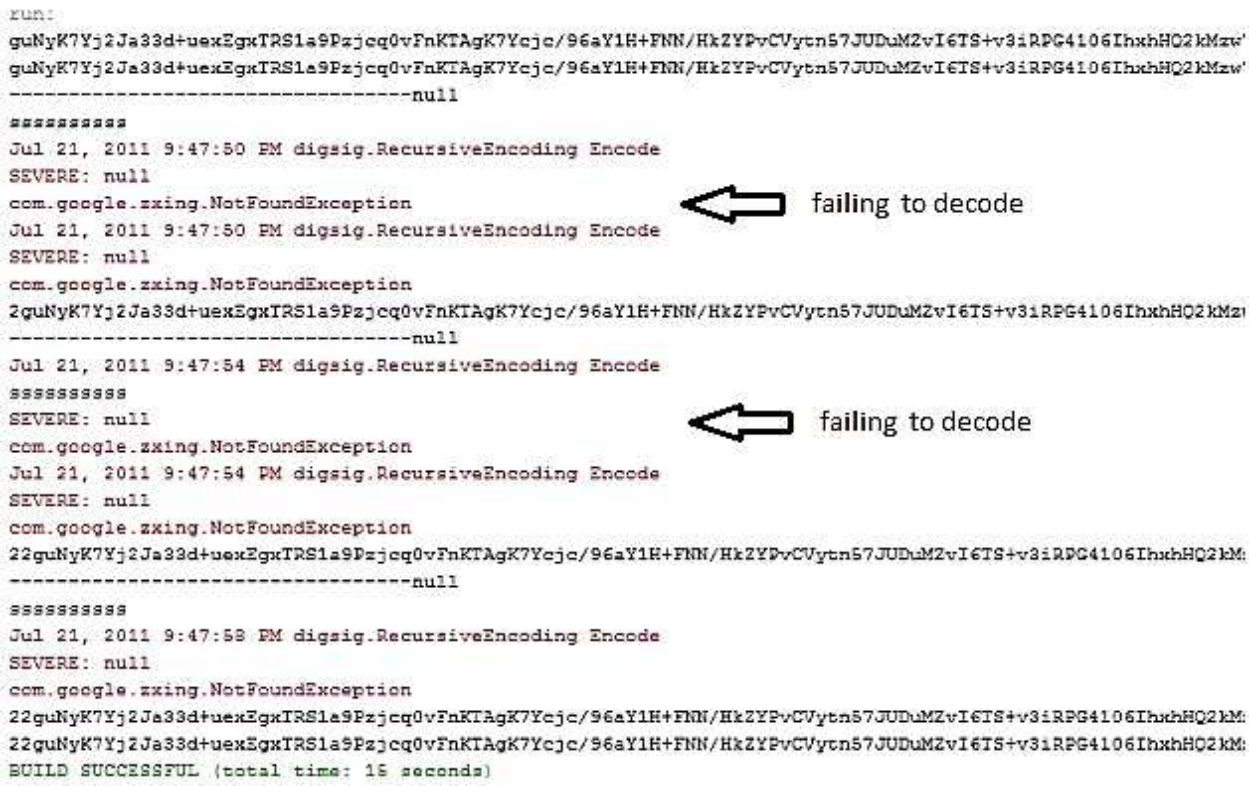


Fig 4.23
Text encoded after padding twice

System Testing tends to affirm the end-to-end quality of the entire system. System test is often based on the functional/requirement specification of the system. Non-functional quality attributes

Found some errors and they were debugged

Conclusion
System is functional no problems were found.

Barcode Scanner

Scanning a hardcopy barcode is a feature in our software.
This means that the user does not have to decode the barcode saved on his computer. Even paper documents with barcode printed on them can be read.

The software of the scanner detects barcodes by default, which means that the user will not exert any effort if he has only his barcode as a hardcopy.

Mobile Checker

Software that decodes the barcode into cipher then decrypts the cipher to retrieve the original data in the document anywhere.
The software is installed on a mobile phone with an integrated camera for portability purposes.
This can be of great use since it can be used by all users.

All mobile checkers decode barcodes, but this one is different only because it also decrypts the data encrypted in a previous stage.

Overall Project Plan

Ser.	Main Activities	Time															
		November 2010				May 2011				June 2011				July 2011			
		W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4
1	Define the problem and analysis																
2	Users approval																
3	Design																
4	Users approval																
5	Developing																
6	Testing																
7	Documentation																
9	Implementation & User Evaluation																
10	Users final approval																

Chapter Five
Conclusion

The solution will improve the security of documents thanks to the encryption and decryption processes previously discussed.

We start with a brief summary of the phases of the two main features of the project,

1. Encoding
    a. User chooses the other party and the document form.
    b. The program encrypts the data entered by the user in a sequence that guarantees the authentication of the sender as well as the confidentiality of data.
    c. The cipher then encoded into a QR barcode.

2. Decoding
    a. User chooses the barcode to be decoded either from his hard disk or from the scanner directory.
    b. The barcode is decoded into the cipher text.
    c. The cipher is then decrypted to retrieve the original data.
    d. The data retrieved will be automatically filled into the HTML form corresponding to the main document.

Using this solution, users will less likely feature the following issues

- Fake documents
- Lost documents
- Document content manipulation
- Partially damaged documents
- Unauthenticated sender/receiver
- Reading difficulties (unclear handwriting)

The program follows the logical sequence needed to issue a legal document.

Further expansion

Future plans

1- A company will be announced to manage and market the solution, a server will be setup and a web site will be developed in order to increase the solution's abilities and flexibility by adding the online features.

2- The users will go to the bank and instead of buying a checkbook he will pay its price and the list of serial numbers will be sent to his account and so his database will be updated by these serial numbers so when he write the check by SUDB the serial number will be written automatically on the check and before he finishes the amount of checks he bought he will receive a notification that he is about to finish the checks to give him time to buy another list of serial numbers.

3- The AES key will be masked and saved in the database with its version. While decoding, SUDB will check every document for the used version of AES and compare it with the latest version used, if it is an old version, SUDB will show an error message asking the user to upgrade it.
The website will allow the user to update the document by decrypting the document with the old AES key and then encrypt it again with the new key and provide the updated document to the user again.

Or, AES's symmetric encryption key can put in the USB flash drive (If there is no main server) and then masked using for example a transposition algorithm before using it in the AES encryption.

Other future expansion plans

- Integrating a key generator for new accounts signup.
- Adding a flash USB burner.
- Acknowledgement of document delivery to the other party using e-mail and SMS.

Notes

- Different time zones are not a problem anymore, since the user can sign the document and send it to the other party to approve and sign anytime with no restrictions.

- Now users can exchange the most confidential documents over the most insecure ways of communication (e-mail, public mail or with a messenger).

- Partially burnt, drowned, or torn barcodes can still be recovered (in case of limited damage only).

References:

1. http://www.onbarcode.com/
2. http://www.barcodelib.com/
3. http://www.denso-wave.com/qrcode/qrfeature-e.html
4. http://www.denso-wave.com/qrcode/aboutqr-e.html
5. http://code.google.com/p/zxing/
6. http://www.w3schools.com/html/html_intro.asp
7. http://www.w3schools.com/js/js_intro.asp
8. William Stallings, Cryptography and Network Security, principles and practices, 4th Edition.
9. [Crypto++]"Crypto++benchmark", http://www.eskimo.com/~weidai/benchmarks.html [Results of comparing tens of encryption algorithms using different settings].
10. [Nadeem2005]Aamer Nadeem et al, "A Performance Comparison of Data Encryption Algorithms", IEEE 2005
11. [Dhawan2002] Priya Dhawan., "Performance Comparison: Security Design Choices," Microsoft Developer Network October 2002.http://msdn2.microsoft.com/en-us/library/ms978415.aspx
12. [Bruce1996] BRUCE SCHNEIER, "Applied Cryptography" , John Wiley & Sons, Inc 1996
13. Ibrahim A. Al-Kadi, "Cryptography and Data Security: Cryptographic Properties of Arabic": Nov 24-27, Vol 2:910-921., 1991.
14. Richard J. Aldrich, GCHQ: The Uncensored Story of Britain's Most Secret Intelligence Agency, HarperCollins July 2010.
15. Oded Goldreich, Foundations of Cryptography, Volume 1: Basic Tools, Cambridge University Press, 2001, ISBN 0-521-79172-3
16. FIPS PUB 197: the official AES standard (PDF file)
17. Whitfield Diffie and Martin Hellman, "New Directions in Cryptography", IEEE Transactions on Information Theory, vol. IT-22, Nov. 1976, pp: 644–654. (pdf).
18. National Institute of Standards and Technology
19. Whitfield Diffie and Martin Hellman, "Multi-user cryptographic techniques" [Diffie and Hellman, AFIPS Proceedings 45, pp109–112, June 8, 1976].
20. David Kahn, "Cryptology Goes Public", 58 Foreign Affairs 141, 151 (fall 1979), p. 153.
21. Levy, Steven (2001). Crypto: How the Code Rebels Beat the Government—Saving Privacy in the Digital Age. Penguin Books. p. 56. ISBN 0-14-024432-8. OCLC 48066852 48846639 244148644 48066852 48846639.