

Progetto di  
Sicurezza Informatica e Internet  
**Progetto A2 - Encrypted 2D Barcodes**

Giovanni Rossi - [gio.rossi.1991@gmail.com](mailto:gio.rossi.1991@gmail.com)  
Pasquale Verlotta - [pasquale.verlotta@gmail.com](mailto:pasquale.verlotta@gmail.com)



# Indice

<b>Indice</b>	<b>I</b>
<b>Elenco delle figure</b>	<b>II</b>
<b>1 Introduzione</b>	<b>1</b>
1.1 Scopo del progetto . . . . .	1
1.2 Tecnologie Utilizzate . . . . .	3
1.2.1 Strumenti di sviluppo . . . . .	3
1.2.2 Linguaggio e librerie utilizzate . . . . .	3
1.3 Le principali tecnologie . . . . .	4
1.3.1 QR-Code . . . . .	5
1.3.2 OCR . . . . .	5
<b>2 Progettazione</b>	<b>7</b>
2.1 Requisiti . . . . .	7
2.2 Architettura . . . . .	8
2.3 Gestione della sicurezza . . . . .	9
2.4 Diagrammi UML . . . . .	9
<b>3 Test</b>	<b>10</b>
3.1 Funzionamento dell'applicazione . . . . .	10
3.2 Test Sperimentali . . . . .	10
<b>4 Conclusioni</b>	<b>11</b>
<b>A Manuale di installazione</b>	<b>12</b>

# Elenco delle figure

2.1 Use Case Diagram . . . . .	8
--------------------------------	---

# Capitolo 1

## Introduzione

### 1.1 Scopo del progetto

Il progetto sviluppato in questi mesi per il corso di “Sicurezza Informatica e Internet” si propone come scopo quello di realizzare un’applicazione che permetta di gestire un documento cartaceo in forma sicura. In particolare questo significa che chi utilizza tale sistema deve essere in grado di

- compilare un foglio con informazioni sensibili;
- cifrare tutto o alcune parti del foglio al fine di poterlo inviare a chiunque voglia;
- decifrare il contenuto di un documento ricevuto da un’altra persona.

Lo scopo ultimo è dunque quello di abilitare l’utente ad inviare ad un gruppo di persone il documento che ha appena creato. Tuttavia durante il trasferimento (funzione che non compete a questa applicazione) può succedere che il documento possa venire alterato in alcune sue parti se non si utilizzano canali sicuri; moltissime truffe avvenute in passato si sono basate su questa vulnerabilità. Dunque uno degli scopi è quello di garantire con opportune tecniche basate su crittografia asimmetrica che il documento cartaceo resti robusto ed inattaccabile qualunque sia il canale di comunicazione che l’utente finale sceglie per l’invio. Nel caso di questo progetto, l’elaborazione riguarda documenti cartacei, quindi non è possibile archiviare le informazioni sensibili in una qualche struttura dati. Dunque la sfida più importante è quella di trasportare insieme al documento stampato le informazioni segrete. Questo apre una serie di problematiche legate alla gestione delle informazioni che sono in formato digitale nella fase di composizione e diventano analogiche nella fase di acquisizione di un documento spedito (che può essere stampato

### 1.1. SCOPO DEL PROGETTO

---

e acquisito più volte durante il percorso - per esempio via fax). La tecnologia migliore per il trasporto dei dati su immagini è rappresentata dal *QR-Code*, un particolare tipo di codice a barre 2D che può contenere al suo interno una discreta quantità di informazioni. I dati possono essere riletti utilizzando algoritmi di riconoscimento robusti che hanno raggiunto nel tempo un'elevata maturità. Nell'ambito di questo progetto i QR-Code sono stati utilizzati sia per memorizzare le informazioni cifrate, sia per apporre una firma digitale al documento intero. Si risponde così alle esigenze di mantenere segreti alcuni contenuti e di rendere il documento robusto alle manipolazioni esterne.

Un'altra funzionalità importante che rientra negli scopi di questa applicazione, è quella di creare una gerarchia tra gli utenti che accedono al servizio. Può capitare, infatti, di voler pubblicare un certo documento e di voler permettere solo a pochi eletti di accedere alle informazioni cifrate. Questo implica che nel sistema si avranno utenti con livelli di privilegio diversi dove ognuno di questi può decifrare e leggere solo le informazioni adeguate al suo livello. In questo caso la soluzione più ovvia potrebbe essere quella di pubblicare versioni diverse dello stesso documento oscurando opportunamente sulle diverse copie le informazioni che si vogliono tenere segrete a ciascun gruppo. Questo processo può risultare lungo, macchinoso e poco sicuro: è possibile, infatti, che qualche copia possa finire al destinatario sbagliato se non si prendono opportune precauzioni. Ci si è quindi occupato di realizzare un sistema dove la cifratura viene effettuata con chiavi speciali (che da ora in poi chiameremo **chiavi di livello**), le quali vengono distribuite solo agli utenti che hanno un **livello di fiducia** (o **Trust Level**) adeguato. In questo caso un Trust Level basso corrisponde ad un basso livello di privilegio, mentre un Trust Level alto garantisce l'accesso a tutte le informazioni cifrate con le chiavi di livello più basso.

Riassumendo quindi, per quanto riguarda le funzionalità principali, il sistema dovrebbe permettere di:

- compilare un documento contenente qualsiasi tipo di informazione;
- cifrare in tutto o in parte il documento appena scritto;
- allegare le informazioni cifrate al documento stesso così che il o i destinatari possano leggerlo;
- firmare il documento in modo tale che il mittente non possa ripudiare la provenienza e i destinatari possano accorgersi di eventuali manomissioni durante il trasferimento;
- specificare diversi livelli di privilegio gerarchici per la lettura delle informazioni.

## 1.2 Tecnologie Utilizzate

Gli obiettivi appena illustrati, danno un'idea delle tante problematiche che l'applicazione deve affrontare. Per questo sono molte le tecnologie che sono state messe in gioco. Di seguito si riportano quelle utilizzate maggiormente.

### 1.2.1 Strumenti di sviluppo

Per quanto riguarda gli strumenti, si è fatto uso di

- **Eclipse Luna** come IDE per la gestione del progetto e dei file sorgente;
- **Apache Maven v2.2** per la gestione delle dipendenze e del ciclo di vita dell'applicazione;
- **Git** per il *versioning* e la condivisione dei sorgenti nel team (in particolare si è usato un repository pubblico<sup>1</sup> su *GitHub* come piattaforma online di condivisione<sup>2</sup>);

### 1.2.2 Linguaggio e librerie utilizzate

In questo progetto è stato utilizzato **Java** come linguaggio di programmazione principale. I motivi di questa scelta ricadono principalmente su:

- il paradigma Object-Oriented che ha permesso l'utilizzo di architectural e creational pattern per risolvere le problematiche più comuni, come Façade, Factory Method, Abstract Factory, DAO e Singleton che sono stati ampiamente utilizzati nello sviluppo;
- l'alta manutenibilità del codice;
- la portabilità su altre piattaforme diverse da quella di sviluppo;
- la grande quantità di librerie presenti a supporto degli obiettivi descritti prima.

Per quanto riguarda invece i singoli moduli del software sviluppato, si sono utilizzati diversi framework e librerie. Per la parte di sicurezza c'è:

---

<sup>1</sup>Per certi versi si è seguito il principio di *Auguste Kerckhoffs* secondo cui “*in un sistema crittografico è importante tener segreta la chiave, non l'algoritmo di crittazione*”.

<sup>2</sup>Link del repository:<https://github.com/WAFcoding/ProgettoSicurezza.git>

### 1.3. LE PRINCIPALI TECNOLOGIE

---

- **JCA/JCE** (Java Cryptographic Architecture/Java Cryptographic Extension) come framework e provider dei principali algoritmi di cifratura, firma e hashing;
- **BouncyCastle** che si integra con JCA/JCE e fornisce un maggior numero di algoritmi di cifratura e firma nonché funzioni di generazione di certificati;

Per il modulo di elaborazione dei documenti:

- **ImageMagik** per la manipolazione delle immagini dei documenti acquisiti per la decodifica;
- **ZXing** per la generazione e lettura di QR-Code (in generale supporta molti tipi di codici a barre sia 1D che 2D);
- **iTextPDF** per la generazione di documenti ad alta risoluzione ed il posizionamento preciso di testo e QR-Code nel documento;
- **Tesseract** ed in particolare il wrapper Java **Tess4J** per il recupero del testo dopo la scansione dei documenti.

Per la parte di persistenza invece si è fatto uso di:

- **MySQL** come database relazionale per il sistema di *provisioning* delle chiavi e la gestione degli utenti;
- **SQLite** per la gestione dei dati locali di ogni singolo utente (sotto forma di database cifrato);
- **Hibernate** come framework per la gestione dei database (MySQL e SQLite), ed in particolare il modulo **ORM** (Object-Relational Mapping).

## 1.3 Le principali tecnologie

Di seguito una breve panoramica dello stato dell'arte delle tecnologie utilizzate maggiormente nello sviluppo di questo progetto. Si analizzeranno in particolar modo il **QR-Code** e l'**OCR**.

#### 1.3.1 QR-Code

Il *QR-Code* (Quick Response Code in virtù del fatto che il codice fu sviluppato per permettere una rapida decodifica del suo contenuto) è un codice a barre bidimensionale, ossia a matrice, composto da moduli neri disposti all'interno di uno schema di forma quadrata. Viene impiegato per memorizzare informazioni generalmente destinate a essere lette tramite un dispositivo mobile. In un solo codice QR possono essere contenuti oltre 7089 caratteri numerici o 4296 alfanumerici. Il QR-Code fu sviluppato nel 1994 dalla compagnia giapponese *Denso Wave* allo scopo di tracciare le componenti di automobili nelle fabbriche della Toyota. Vista la capacità del codice di contenere più dati di un codice a barre venne in seguito utilizzato per la gestione delle scorte da diverse industrie. Tuttavia essendo un codice che deve essere riportato su un documento cartaceo, può subire dei danni che possono provocare degli errori nella seguente lettura. Per questo viene utilizzato il codice Reed-Solomon per la rilevazione e correzione d'errore che permette di ricostruire i dati persi, ripristinando, durante la decodifica, fino al 30% delle informazioni codificate.

#### 1.3.2 OCR

La parola *OCR* (Optical Character Recognition) descrive una classe di sistemi di riconoscimento ottico dei caratteri. Gli OCR sono programmi dedicati alla conversione di un'immagine contenente testo, solitamente acquisite tramite scanner (come nel caso di questo progetto), in testo digitale modificabile con un normale editor. Il testo può essere convertito in formato ASCII semplice, Unicode o, nel caso dei sistemi più avanzati, in un formato contenente anche l'impaginazione del documento. L'OCR è un importante campo di ricerca dell'intelligenza artificiale, della visione artificiale e del pattern recognition, legati al riconoscimento delle immagini. I sistemi OCR per funzionare correttamente richiedono una fase di "addestramento". Durante questa fase al sistema vengono forniti degli esempi di immagini col corrispondente testo in formato ASCII o simile in modo che gli algoritmi si possano calibrare sul testo che usualmente andranno ad analizzare. Gli ultimi software di OCR utilizzano algoritmi in grado di riconoscere i contorni e in grado di ricostruire oltre al testo anche la formattazione della pagina. Gli OCR possono essere usati per riconoscere:

- caratteri stampati, per i quali gli algoritmi noti hanno un tasso di errore del 1%;



### *1.3. LE PRINCIPALI TECNOLOGIE*

---

- scrittura a mano libera, problema tutt'altro che risolto, per la quale gli algoritmi hanno tassi di accuratezza dell'80% - 90%;
- caratteri in corsivo, che è un campo in fase di studio, per il quale si hanno tutt'oggi risultati poco soddisfacenti a livello di accuratezza.

# Capitolo 2

## Progettazione

In questo capitolo verranno analizzate in dettaglio le scelte progettuali che hanno permesso la realizzazione del sistema software, considerando le limitazioni riscontrate nell'architettura del sistema stesso. Dopodiché si analizzerà, attraverso la rappresentazione dei diagrammi **UML**, la definizione delle fasi di sviluppo, ossia la progettazione delle funzionalità correnti dell'applicazione.

### 2.1 Requisiti

Come primo aspetto della progettazione si analizzeranno adesso i requisiti del sistema. Innanzitutto si deve specificare che l'intero sistema è pensato per essere utilizzato in un ambito amministrativo dove gli utenti possono essere per esempio i dipendenti di un'azienda. Questo vuol dire che ci si trova in un sistema chiuso in cui le persone che usano il sistema sono note. Tuttavia è possibile che nel tempo i dipendenti si licenzino oppure che ne sopraggiungano di nuovi, quindi deve essere presente una procedura che consenta di inserire nuovi utenti ed eventualmente di rimuovere quelli esistenti. In questo sistema dove gli utenti sono noti, immaginiamo la presenza di un utente **amministratore** che agisce come supervisore ed è l'unico ad avere la facoltà di abilitare gli utenti all'utilizzo del servizio nonché all'assegnazione del *Trust Level*.

Di seguito viene mostrato il diagramma dei casi d'uso che rappresenta una vista generale delle funzionalità del sistema.

## 2.2. ARCHITETTURA

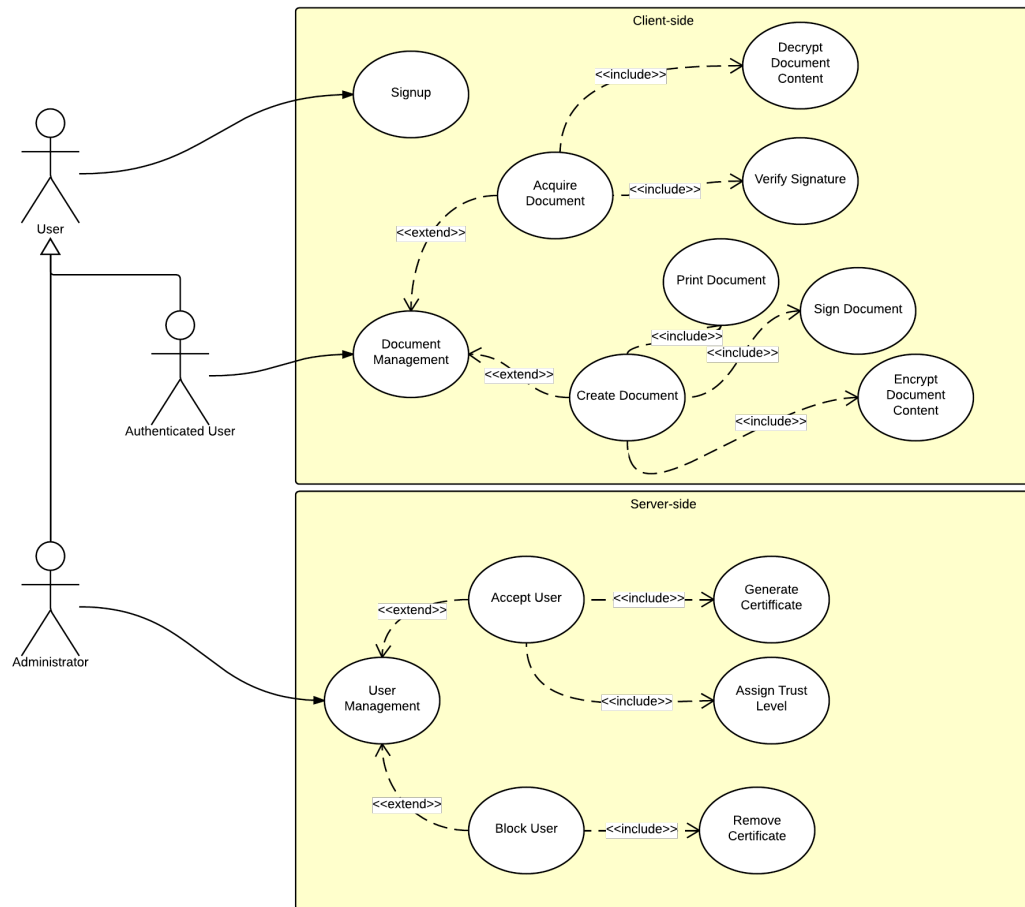


Figura 2.1: Diagramma dei casi d'uso del sistema software realizzato

## 2.2 Architettura

Per questa applicazione si è deciso di sviluppare il tutto con un architettura di tipo client-server.

## **2.3 Gestione della sicurezza**

## **2.4 Diagrammi UML**

# Capitolo 3

## Test

### 3.1 Funzionamento dell'applicazione

### 3.2 Test Sperimentali

**Capitolo 4**

**Conclusioni**

# Appendice A

## Manuale di installazione