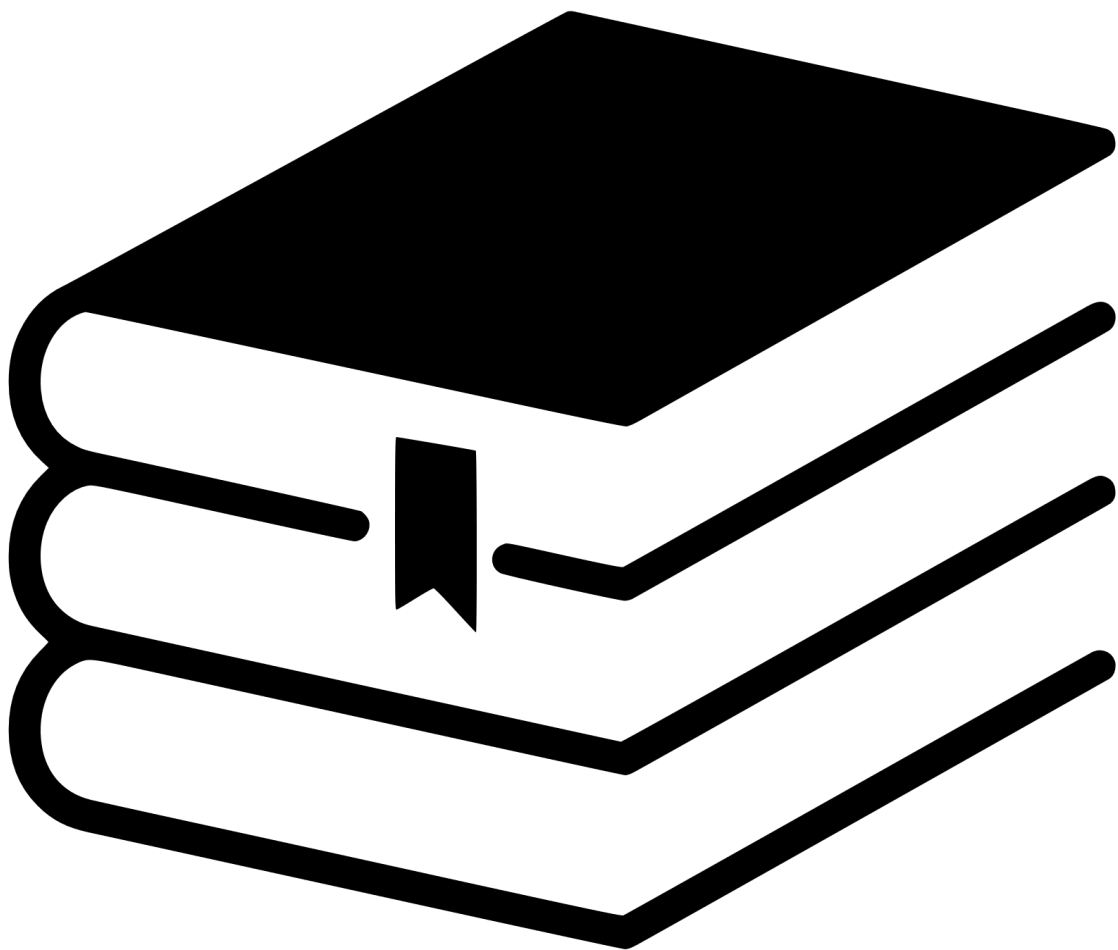


REQUIREMENTS ANALYSIS DOCUMENT

OBJECT ORIENTED SOFTWARE DESIGN

UNIVERSITA' DEGLI STUDI DELL'AQUILA

BIBLIOTECA DIGITALE



ANNO 2017/2018

Index

1. Introduzione
 - 1.1 Scopo del sistema
 - 1.2 Definizioni, acronimi, abbreviazioni
 - 1.3 Fonti
2. Sistema corrente
3. Sistema proposto
 - 3.1 Panoramica
 - 3.2 Requisiti funzionali
 - 3.3 Requisiti non funzionali
 - 3.3.1 Usabilità
 - 3.3.2 Affidabilità
 - 3.3.3 Implementazione
 - 3.3.4 Packaging
 - 3.3.5 Cenni legali
 - 3.4 Modelli del sistema
 - 3.4.1 Scenari
 - 3.4.2 Modello use case
 - 3.4.3 Object model
 - 3.4.4 Dynamic Model
 - 3.4.5 Interfaccia utente

1. Introduzione

Questo progetto è stato realizzato da:

Antokhi Stanislav, Mat. 244295

D'Angelo Andrea, Mat. 248817

Diociaiuti Stefano, Mat. 246856

Zappacosta Francesco, Mat. 249168

Per il corso 2017/2018 di Object Oriented Software Design all'Università degli studi dell'Aquila.

L'intera documentazione e codice si può trovare nel repository di GitHub:
<https://github.com/dangeloandrea14/Project>.

Come punto di riferimento, sia per la costruzione di questo documento che per la restante documentazione, abbiamo preso il libro *“Object-Oriented Software Engineering. Using UML, Patterns and Java”* di Bernd Bruegge e Allen H. Dutoit.

Questo progetto è stato realizzato come esame finale del corso di Object Oriented Software Design, e la sua funzione è quella di fornire un servizio di biblioteca digitale per la visualizzazione online di manoscritti del patrimonio bibliografico antico della città dell'Aquila.

1.1 Scopo del Sistema

Lo scopo del progetto è la realizzazione di una biblioteca digitale formata da manoscritti per un totale di 60.000 carte (ms. sec. XV-XIV) contenenti memorie storiche della città dell'Aquila, consentendo la consultazione dei manoscritti, che devono essere digitalizzati. Il sistema deve quindi sia permettere l'upload e la digitalizzazione delle varie opere, che la loro consultazione da parte degli utenti.

1.2 Definizioni, acronimi, abbreviazioni

Con "manoscritto" intendiamo qualsiasi pubblicazione voglia essere caricata o letta sul sistema.

1.3 Fonti

Come fonte per tutti i requisiti abbiamo utilizzato la specifica del progetto pubblicata. Come riferimento per tutta la strutturazione del progetto e della documentazione, invece, abbiamo utilizzato il libro di testo consigliato.

2. Sistema corrente

Non c'è alcun sistema correntemente progettato per questo dominio di applicazione, e non è dunque presente alcun punto di riferimento a priori per le scelte di design che sono state prese. Queste scelte sono accuratamente descritte nella documentazione e sono state prese in gruppo dopo accurate riflessioni e costruzione di scenari che hanno portato alla struttura del modello Use case.

La costruzione degli scenari è stata fatta in modo tale da essere il più coerente possibile con il dominio dell'applicazione descritto dalla specifica.

3. Sistema proposto

3.1 Panoramica

Il sistema proposto garantisce un ruolo specifico ad ogni utente con dei relativi permessi gestibili dall'amministratore, che è unico. I ruoli e i permessi sono gestiti da un database interno al sistema, dove vengono memorizzati anche gli utenti registrati, le pubblicazioni e gli scan.

Solo gli utenti registrati come Uploader possono accedere al sistema di caricamento dei manoscritti, che è invisibile ai normali utenti. Il sistema gerarchico di ruoli, quindi, garantisce ad ogni utente solo una parte delle funzionalità del sistema, per mantenere il sistema sicuro e funzionante.

3.2 Requisiti funzionali

Il sistema deve:

- Permettere agli utenti di registrarsi
- Permettere agli utenti di fare il login
- Permettere agli utenti di ricercare manoscritti tramite metadati
- Permettere la visualizzazione dei manoscritti, con annessa trascrizione
- Consentire la visualizzazione del proprio profilo personale
- Permettere di scaricare le opere a determinati utenti
- Permettere di candidarsi come trascrittori
- Gestire l'upload delle scansioni delle pagine
- Permettere la revisione degli upload --

- Avere un editor di testo integrato per la trascrizione degli scan
- Consentire la gestione in back-end del sistema all'amministratore
- Assegnare scan da trascrivere ai trascrittori --
- Revisionare le trascrizioni concluse --
- Pubblicare trascrizioni e opere
- Gestire i livelli dei trascrittori

Tutti questi sono i requisiti funzionali descritti dalla specifica del progetto.

3.3 Requisiti non funzionali

3.3.1 Usabilità

Il sistema è indirizzato al più ampio spettro di utenti finali possibile, allo scopo di condividere la conoscenza dei manoscritti. Funzionalmente, quindi, il sistema deve essere il più semplice possibile da utilizzare, con label chiare ed efficaci, per ingaggiare anche il pubblico meno esperto.

3.3.2 Affidabilità

E' importante che utenti non autorizzati non possano avere accesso a risorse riservate o al management in back-end del sistema. Un accesso non autorizzato potrebbe portare alla perdita di informazioni fondamentali.

3.3.3 Implementazione

L'implementazione deve essere fatta tramite Java, JavaFX e il software Scene Builder.

3.3.4 Packaging

Il packaging deve essere realizzato seguendo il modello MVC (Model – View – Controller). E' importante che il codice sia ben diviso in sottosistemi autonomi che comunicano tra di loro. I dettagli esatti del packaging sono esplicitati nella specifica.

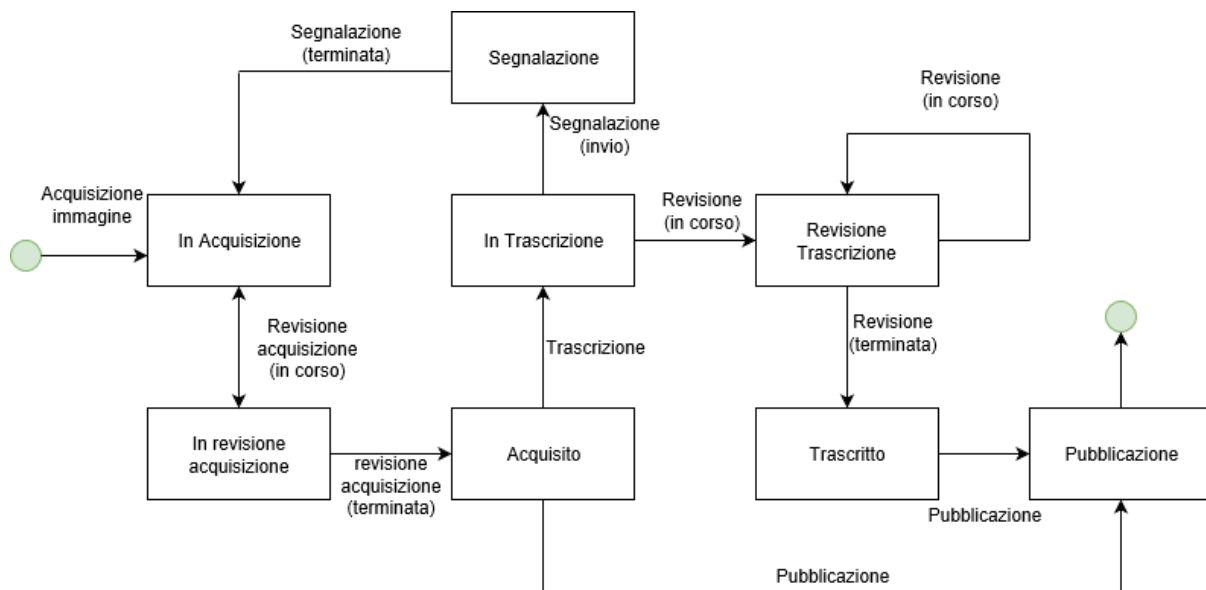
3.3.5 Cenni legali

I diritti legali di manutenzione e gestione dei manoscritti sono del tutto riservati all'azienda che gestisce. Il nostro software è stato sviluppato sotto loro autorizzazione.

3.4 Modelli del Sistema

3.4.1 Scenari

Per la costruzione degli scenari ci siamo basati su questo diagramma, che è una versione modificata e più completa di quello fornito con le specifiche.



Gli scenari basati su questo diagramma sono quelli relativi alla pubblicazione e trascrizione dei manoscritti. Andiamo ad illustrare in questa documentazione solo alcuni degli scenari costruiti:

Primo scenario:

Nome scenario: Pubblicazione immagine

Attori partecipanti: Berta: Uploader, Mario: RevisoreUpload

Flusso di eventi:

1. Berta ha un manoscritto e decide di fare la scansione della pagina 33 del manoscritto. Avvia allora la funzionalità di Upload a cui ha accesso in quanto uploader.
2. Berta carica il file attraverso la funzione di upload. Inserisce i dati relativi al manoscritto di cui sta facendo la scansione e il numero della pagina.
3. Mario vede il file. Dato che la qualità è accettabile e il testo è leggibile, decide di accettare la scansione, che va quindi ad arricchire la collezione di scansioni per quel manoscritto.

Secondo scenario:

Nome scenario: Iscrizione come Trascrittore

Attori partecipanti: Bob: Utente, John: Amministratore

Flusso di eventi: 1. Bob, utente utilizzatore del servizio da tempo, decide di voler aiutare la causa di Condividere conoscenza della città dell'Aquila e aiutare con la trascrizione degli scan. Per fare questo, attiva la funzionalità "Proponiti come trascrittore" dal pulsante che si trova nel suo profilo.

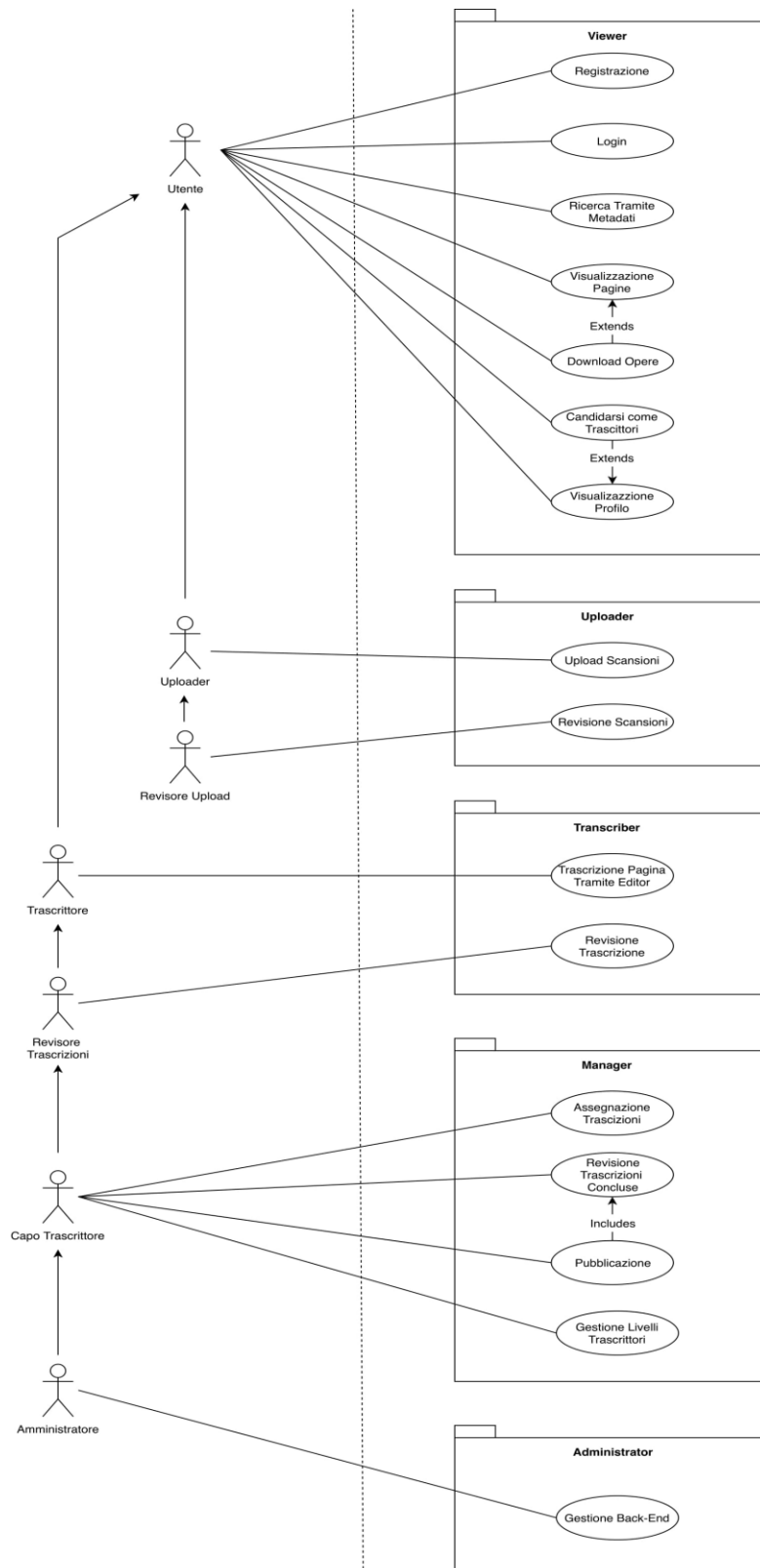
4. John, l'amministratore, vede che Bob si è proposto come trascrittore. Attraverso i dati in suo possesso, decide che Bob è un ottimo utente e di conseguenza il suo aiuto sarebbe utile, e lo accetta come trascrittore.
 5. Bob ora può accedere alle funzionalità esclusive per i trascrittori.
-

3.4.2 Modello use case

Il modello use case è stato costruito a partire dalla generalizzazione degli scenari che abbiamo costruito. E' anche specificato nello schema la limitazione ai confini del sistema. Alcune note sul diagramma use case UML:

- 1) Il database utilizzato nel sistema è stato considerato come interno e, di conseguenza, non un attore. Questa è una scelta di design: se avessimo considerato un server o database proprietario dell'organizzazione, lo avremmo trattato come attore.
- 2) Volendo si potrebbero aggiungere diversi use case per estendere use case esistenti in caso di eccezioni o comportamenti imprevisti. Dato che lo scopo del modello use case è quello di essere il più chiaro possibile anche per i clienti, si è voluto evitare di introdurre ulteriore complessità se l'use case estendeva un solo altro use case.

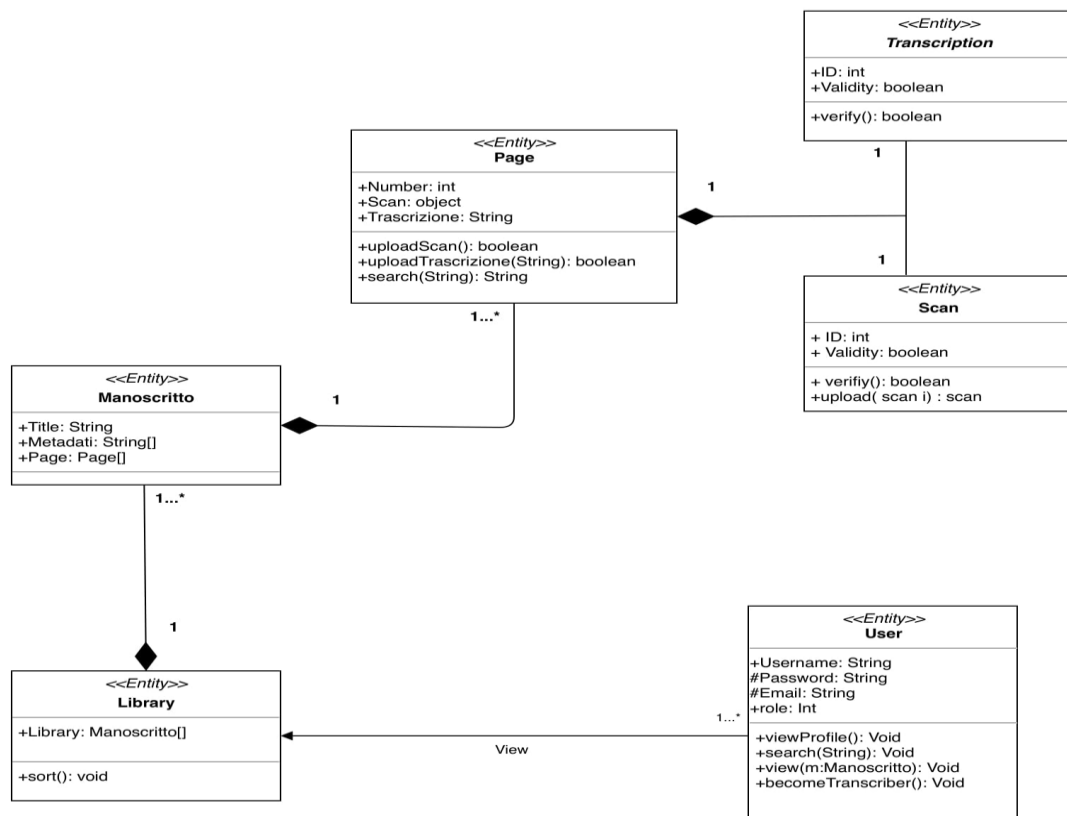
Il risultato del nostro lavoro di modellazione e Requirements Elicitation è il seguente:



Il file originale può essere trovato nella documentazione originale.

3.4.3 Object Model

Il modello a oggetti è un modello per trattare un problema attraverso la tipica rappresentazione Object-Oriented.



Quello in figura è il modello di Dominio con le classi entity richiesto dalla specifica 1.3 della specifica. L'analisi delle relative classi entity, boundary e controller può essere trovata nelle specifiche allegate.