“Objecte Oriented Programming”

Course

a.a. 2015-2016

MEMBRI DEL TEAM

Studente	Matricola	E-mail
Angelini Marco	228613	marco.angelini1994@gmail.com
Amadio Marco	230406	mark_ap@hotmail.it

PARTE 1

1. REQUIREMENTS

1.1) DOCUMENTO DEI REQUISITI

Il nostro progetto consiste nella creazione di una biblioteca digitale di testi e studi che contribuiscono alla formazione della cultura all'interno dell'Università degli Studi dell'Aquila.

Per popolare la nostra biblioteca sarà necessario compiere tre processi fondamentali:

- Digitalizzazione;
- Trascrizione;
- Pubblicazione;

Digitalizzazione

Questo primo processo consiste nell'acquisizione in formato digitale del manoscritto attraverso scanner planetari.
La fase d'acquisizione è gestita da un attore opportuno (*Acquisitore*).
Al processo d'acquisizione segue una fase di revisione delle immagini catturate, anch'essa gestita da un determinato attore (*Revisore Acquisizioni*).

Trascrizione

Nella seconda fase, il manoscritto acquisito deve essere trasformato, mediante operazioni di trascrizione, in formato TEI (Text Encoding Initiative).
Il processo di trascrizione ha come protagonisti due attori; il primo (*Trascrittore*) si occuperà della vera e propria trasformazione in formato TEI, mentre il secondo (*Revisore Trascrizioni*) avrà il compito di verificare la correttezza del testo.

Pubblicazione

Questa terza e ultima fase consiste nel pubblicare i manoscritti e renderli accessibili agli utenti avanzati.
Per ciascun manoscritto verranno pubblicate sia le immagini acquisite, sia i corrispettivi testi TEI.
I manoscritti, una volta digitalizzati e superata la fase di revisione delle immagini, vengono
pubblicati sul sistema.
Le corrispondenti trascrizioni sono pubblicate successivamente, dopo la validazione della stessa..

ATTORI RISULTANTI DEL SISTEMA

- Acquisitore
- Revisore Acquisizioni
- Trascrittore
- Revisore Trascrizioni
- Utente base
- Utente avanzato
- Amministratore

ACQUISITORE

L'acquisitore ha il compito in primis di acquisire le immagini relative ai manoscritti tramite scanner planetari; in seguito provvederà a caricarle per renderle disponibili al revisore delle acquisizioni.

REVISORE ACQUISIZIONI

Il revisore ha il compito di controllare le immagini acquisite; in caso di correttezza provvederà alla pubblicazione delle stesse nel sistema, altrimenti avviserà l'acquisitore riguardo l'errore trovato.

TRASCRITTORE

Il compito principale del trascrittore è quello di generare file TEI corrispondenti ai manoscritti. Successivamente caricherà il file risultante per renderlo accessibile all'opportuno revisore.

REVISORE TRASCRIZIONI

Questo attore si occuperà della revisione dei file trascritti; in caso di correttezza pubblicherà i file nel sistema, altrimenti riferirà le modifiche da apportare.

UTENTE BASE

L'utente base corrisponde ad un utente generico non ancora registrato nel sistema.
Sarà un attore con funzionalità limitate: potrà infatti accedere solamente all'elenco dei titoli delle opere.

UTENTE AVANZATO

L'utente avanzato corrisponde ad un utente registrato nel sistema e quindi potrà usufruire dei manoscritti presenti in esso.

AMMINISTRATORE

E' una persona fisica che ha accesso a qualsiasi fase di digitalizzazione coordinandone al meglio le fasi.

FUNZIONALITA' RISULTANTI DEGLI ATTORI

FUNZIONALITA'	ATTORE	BREVE DESCRIZIONE
Acquisizione Immagini	Acquisitore	Acquisisce immagini tramite scanner planetari
Caricamento Immagini	Acquisitore	Rende le immagini revisionabili dal Revisore Acquisizioni
Visualizzazione Acquisizioni	Revisore Acquisizioni	Visualizza singolarmente le immagini per la revisione
Errore Acquisizione	Revisore Acquisizioni	Comunica l'errore di scan all'acquisitore, che provvederà a ripetere l'acquisizione
Pubblicazione Acquisizioni	Revisore Acquisizioni	Rende pubblici i manoscritti scannerizzati
Trascrizione Opera	Trascrittore	Creazione file TEI dell'opera
Caricamento Trascrizioni	Trascrittore	Rende i file TEI revisionabili dal Revisore Trascrizioni
Visualizzazione Trascrizioni	Revisore Trascrizioni	Visualizza i file TEI per la revisione
Errore Trascrizione	Revisore Acquisizioni	Comunica l'errore di trascrizione al Trascrittore
Pubblicazione Trascrizioni	Revisore Trascrizioni	Rende pubblici i manoscritti in formato testo
Registrazione	Utente Base	Iscrizione al servizio di biblioteca digitale

Visualizzazione titoli	Utente Base	Può visualizzare solo l’elenco dei titoli delle opere
Registrazione	Utente Avanzato	Iscrizione al servizio di biblioteca digitale
Login	Utente Avanzato	Accesso al portale
Visualizzazione opere	Utente Avanzato	Può visualizzare le opere in maniera completa
Creazione opera	Amministratore	Inserisce nel sistema il titolo dell’opera
Elimina Opera	Amministratore	Elimina l’opera dal sistema, con le relative immagini e trascrizioni

2. SYSTEM DESIGN

2.1) MODELLO DELL'ARCHITETTURA SOFTWARE
Soffermandoci sulle specifiche e le funzionalità del progetto, abbiamo optato per un modello software CLIENT-SERVER.

Un sistema client-server indica un'architettura di rete nella quale genericamente un computer client o terminale si connette ad un server per la fruizione di un certo servizio, quale ad esempio la condivisione di una certa risorsa hardware/software con altri client, appoggiandosi alla sottostante architettura protocollare.

Essendo il nostro sistema progettato per produrre servizi web abbiamo adoperato questa architettura. La funzionalità primaria del sistema messa a disposizione dal server è la DIGITALIZZAZIONE, che consiste quindi in tutti i processi di acquisizione, trascrizione, revisione ed infine pubblicazione dei manoscritti.

2.2) DESCRIZIONE DELL'ARCHITETTURA

Abbiamo deciso di descrivere la nostra architettura basandoci su un diagramma UML: il component diagram.
Questo diagramma ha lo scopo di rappresentare la struttura interna del sistema software modellato in termini dei suoi componenti principali e delle relazioni fra di essi. Per componente si intende una unità software dotata di una precisa identità, nonché responsabilità e interfacce ben definite.

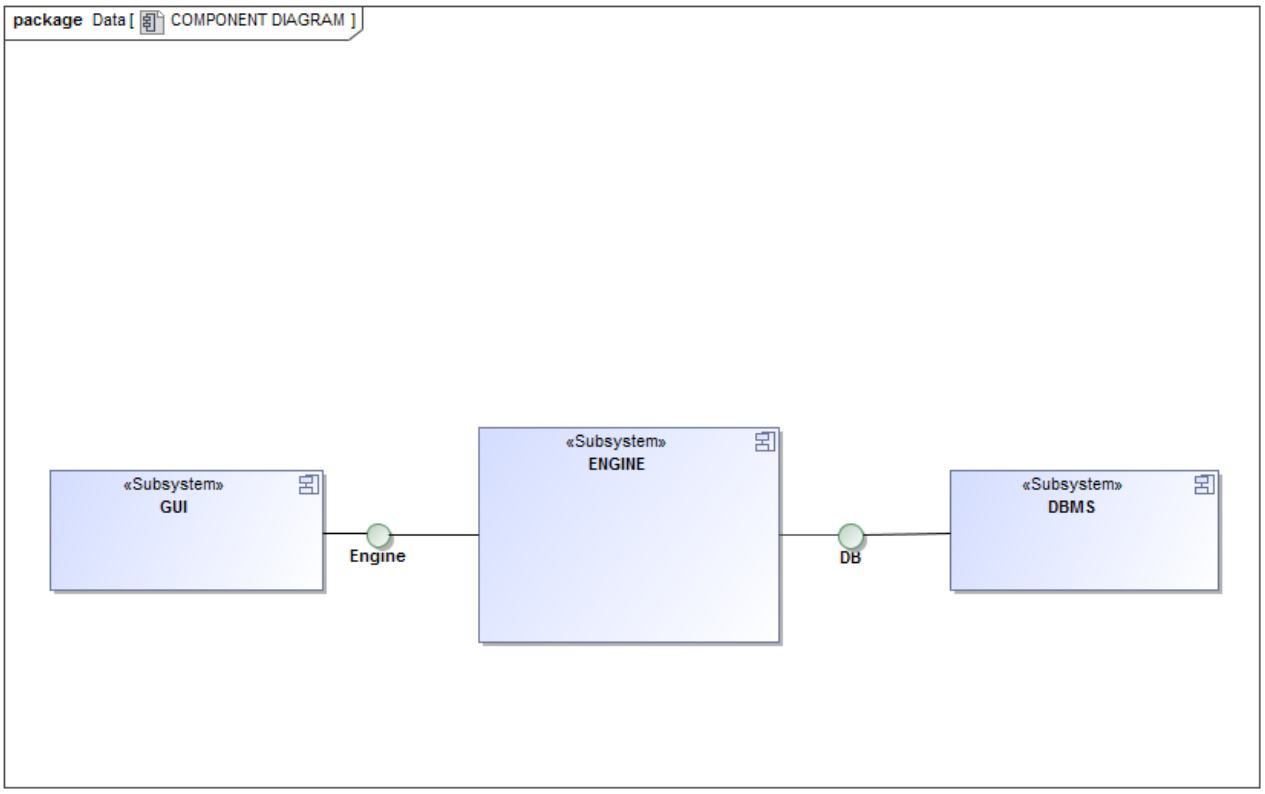
Come si può notare nella figura sottostante, esso mostra la presenza di 3 componenti principali:

- GUI
- ENGINE
- DBMS

La **GUI** è l'interfaccia grafica del nostro sistema, che, attraverso l'uso di opportuni pulsanti, caselle di testo e selezioni, consente all'UTENTE di interagire con il sistema; potrà inoltre visualizzare messaggi di successo o di errore in base al risultato delle operazioni effettuate.
La GUI viene graficamente rappresentata come un sotto-sistema all'interno del boundary: l'UTENTE, infatti, si relaziona solo con la GUI ed essa, a sua volta, si relaziona con ENGINE.

L'**ENGINE** è il motore del nostro sistema; in esso sono racchiuse tutte le fasi del processo di digitalizzazione dei manoscritti, ossia la fase di acquisizione, trascrizione e pubblicazione.
Nella fase di acquisizione, l'acquisitore ha il compito di caricare le immagini su un database, che mette in comunicazione l'acquisitore e il revisore.
Quest'ultimo ha il compito di controllare se l'immagine può essere accettata: tale controllo verrà effettuato usando un pulsante CONFERMA. Nel momento in cui tutte le immagini relative ad un opera verranno confermate, il revisore provvederà alla pubblicazione del manoscritto.
Nella fase di trascrizione, avviene un processo simile al precedente; il trascrittore infatti, dopo aver creato file TEI relativi all'opera, si relazionerà con il revisore, che provvederà a controllare la correttezza della stessa.
L'opera verrà pubblicata solo dopo che il revisore avrà confermato tutte le trascrizioni.
Come descritto in precedenza, la fase di pubblicazione è un compito che spetta ai revisori delle opere.
L'ENGINE si relaziona con il DBMS.

Il **DBMS** offre metodi opportuni per permettere l'interazione con il database, all'interno del quale vengono salvate tutte le immagini e le trascrizioni di ciascun opera, al fine di renderle accessibili e utilizzabili agli utenti.


package Data [COMPONENT DIAGRAM]
«Subsystem»
GUI
Engine
«Subsystem»
ENGINE
DB
«Subsystem»
DBMS

2.3) DESCRIZIONE DELLE SCELTE E STRATEGIE ADOTTATE

Realizzazione Component Diagram
La realizzazione del Component Diagram ci ha permesso di individuare tre sotto sistemi: GUI, ENGINE e DBMS.
La scelta di tale suddivisione è dipesa dal seguente ragionamento: la GUI consente all'UTENTE di interagire con il sistema, mettendo a disposizione opportuni pulsanti, caselle di testo e selezioni per le relative funzionalità.
Una volta gestito ciò, è necessaria la presenza di un sotto-sistema che faccia da vero e proprio motore: a tal proposito abbiamo inserito il sotto-sistema ENGINE.
Quest'ultimo, per poter effettuare tali funzionalità, deve interagire con il DBMS, che permette di interrogare il database al fine di poter accedere ai dati richiesti. Tale suddivisione ci consente di ridurre la complessità del sistema.

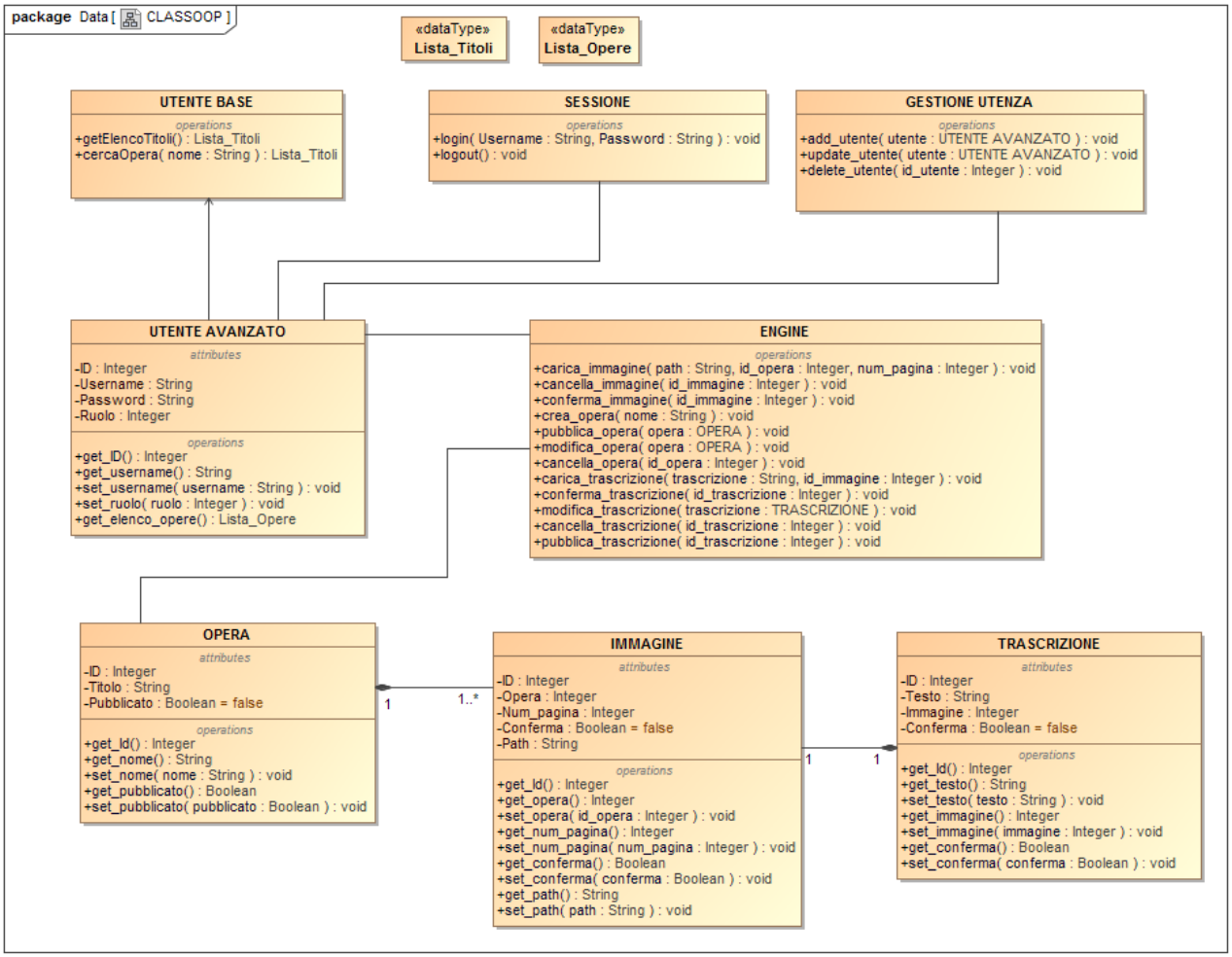
Ricerca dell'opera
Abbiamo introdotto la funzionalità di ricerca dell'opera. All'utente si presenterà come una text-box nella quale inserirà del testo (una parola chiave ad esempio) in riferimento al titolo dell'opera.
Il compito principale di questa text-box è quello di verificare se un opera è presente o meno nel database.
Rende inoltre più facile l'accesso ad un opera da parte degli utenti.
Questa funzionalità è resa accessibile sia agli utenti base che a quelli avanzati.

Pulsante conferma
Come descritto nella specifica, i revisori delle immagini e delle trascrizioni hanno il compito di verificare la loro correttezza.
Abbiamo quindi deciso di introdurre un campo di conferma come funzionalità dei revosori; campo settato inizialmente a false.
Qualora l'oggetto da revisionare, che sia un immagine o un file TEI, dovesse rispettare tutti i requisiti e gli standard il revisore imposterà il campo a true.
Nel momento in cui tutti i file saranno confermati, i revisori provvederanno alla pubblicazione.
Se dopo la fase di revisione qualche immagine o trascrizione continuerà ad avere conferma settato a false, allora l'acquisitore e il trascrittore dovranno procedere con le opportune correzioni, ricaricando quanto modificato e facendolo revisionare nuovamente.

Utenti ed interfaccia GUI
Per consentire agli utenti di interfacciarsi con il sistema, utilizziamo l'interfaccia GUI. Le funzionalità saranno diverse a seconda se il sistema è utilizzato da un utente base o da un utente avanzato.
L'utente base avrà la possibilità di visualizzare solo il titolo delle opere pubblicate, ricercare le opere per titolo attraverso una barra di ricerca nell'interfaccia ed eventualmente registrarsi.
Nella fase di registrazione, un generico utente base inserirà i propri dati personali, creando un account che gli consentirà di loggarsi.
Una volta registrato, l'utente sarà classificato come utente avanzato e avrà uno username e una password che digiterà ogni volta che intenderà accedere al sistema. Una volta loggato, l'utente avanzato può visualizzare le opere pubblicate, ricercare le opere per titolo e, a differenza di un utente base, può anche accedervi, e quindi leggerle.

Utenti speciali
Acquisitori, trascrittori e revisori sono dei particolari tipi di utenti avanzati.
Vengono registrati da un amministratore che attribuirà loro le credenziali per poter accedere al sistema e i ruoli che questi dovranno ricoprire.

3. SOFTWARE / OBJECT DESIGN



Per la rappresentazione dell' object design relativo al nostro sistema ci siamo basati sullo schema UML in figura: il Class Diagram.
Questo diagramma fornisce una vista strutturale del sistema in termini di Classi e Relazioni fra esse.

Nel nostro schema abbiamo individuato le seguenti classi:

- **UTENTE BASE**
- **UTENTE AVANZATO**
- **OPERA**
- **IMMAGINE**
- **TRASCRIZIONE**
- **ENGINE**
- **SESSIONE**
- **GESTIONE UTENZA**

Classe **UTENTE BASE:**

Metodi

- **getElencoTitoli(),** restituisce l'elenco dei titoli delle opere salvate nel sistema,
- **cercaOpere(nome: String),** metodo che ci permette di fare una ricerca del titolo dell'opera.

Classe **UTENTE AVANZATO**

Attributi

- **ID,** di tipo integer, identifica l'utente,
- **Username,** stringa che viene utilizzata per effettuare il login al nostro sistema e riconoscere l'utente,
- **Password,** stringa che viene utilizzata per effettuare il login al nostro sistema,
- **Ruolo,** rappresenta il tipo che l'utente avanzato può assumere (es: ACQUISITORE, TRASCRITTORE, ecc.)

Metodi

- **get_ID(),** restituisce l'ID dell'utente avanzato;
- **get_username() e set_username(username: String),** ci consentono rispettivamente di restituire l'username e di modificarlo con la stringa "username" che gli passiamo come parametro;
- **set_ruolo (ruolo: integer),** ci consente di modificare il ruolo dell'utente, tramite l'integer passato come parametro;
- **get_elenco_opere(),** questo metodo ci restituisce l'elenco completo delle opere registrare nel sistema.

Classe **SESSIONE**

Metodi

- **login (username: String, password: String),** form che ci permette di entrare nel sistema ed eseguire servizi in base al ruolo dell'utente;
- **logout(),** metodo che ci permette di uscire dal sistema come utente avanzato.

Classe **GESTIONE UTENZA**

Metodi

- **add_utente (utente: UTENTE AVANZATO),** metodo che ci permette di aggiungere un utente al sistema;
- **update_utente (utente: UTENTE AVANZATO),** metodo che ci permette di aggiornare un utente, inserendo i dati passati attraverso il parametro utente;
- **delete_utente (id_utente: Integer),** metodo che ci permette di eliminare l'utente che ha per ID il parametro passato.

N.B.: tutti i metodi della classe **GESTIONE UTENZA,** possono essere svolti *esclusivamente* dall'AMMINISTRATORE.

Classe **OPERA**

Attributi

- **ID,** di tipo integer, è un identificatore univoco dell'opera;
- **Titolo,** di tipo String, riguarda il titolo dell'opera;
- **Pubblicato,** di tipo boolean, ci permette di verificare se l'opera è stata confermata o meno dall'opportuno revisore;

Metodi

- **get_id(),** ci permette di restituire l'id dell'opera;
- **get_nome() e set_nome(nome: String),** ci permettono, rispettivamente, di restituire il nome e modificarlo con il "nome" passato come parametro;
- **get_pubblicato() e set_pubblicato (pubblicato: Boolean),** il primo metodo ci permette di vedere se l'opera è stata pubblicata o no, attraverso la restituzione del boolean;
  il secondo metodo invece, ci permette di modificarlo;

Classe **IMMAGINE**

Attributi

- **ID,** di tipo Integer, identifica l'immagine in maniera univoca;
- **Opera,** di tipo Integer, sta ad indicare a quale opera si riferisce;
- **Num_pagina,** di tipo Integer, identifica il numero di pagina di una rispettiva immagine;

- **Conferma,** sta ad indicare se la pagina è stata confermata o meno dai rispettivi revisori;
- **Path,** di tipo String, fa riferimento al percorso dell'immagine nel sistema.

Metodi

- **get_Id(),** restituisce l'id dell'immagine;
- **get_opera() e set_opera (id_opera: Integer),** il primo metodo ci permette di restituire l'id dell'opera di riferimento, invece, con il secondo possiamo modificarlo con "id_opera" che passiamo come parametro;
- **get_num_pagina() e set_num_pagina(num_pagina: Integer),** ci permettono, rispettivamente, di restituire il numero di pagina dell'immagine, e di modificarlo con il "num_pagina" che passiamo come parametro;
- **get_conferma() e set_conferma(conferma: Boolean),** con il primo metodo ci facciamo restituire se l'immagine è stata confermata o no, mentre con il secondo modifichiamo tale scelta;
- **get_path e set_path (path: String),** ci permettono, rispettivamente, di restituire il percorso dell'immagine e di modificarlo tramite il parametro "path";

Classe **TRASCRIZIONE**

Attributi

- **ID,** di tipo Integer, identifica la trascrizione in modo univoco;
- **Testo,** di tipo String, corrisponde al testo trascritto dell'opera;
- **Immagine,** di tipo Integer, si riferisce all'ID dell'immagine che corrisponde alla trascrizione;
- **Conferma,** di tipo Boolean, ci indica se l'immagine è stata confermata o meno;

Metodi

- **get_Id(),** restituisce l'ID della trascrizione;
- **get_testo() e set_testo(testo: String),** il get restituisce il testo della trascrizione, mentre il set lo modifica con la variabile "testo" che passiamo come parametro;
- **get_immagine() e set_immagine (immagine: Integer),** il primo metodo restituisce l'ID dell'immagine corrispondente alla trascrizione, mentre il secondo modifica l'ID dell'immagine.
- **get_conferma() e set_conferma (conferma: Boolean),** il primo metodo ci informa sullo stato della conferma: se true o false; il secondo ci permette di settarlo.

Classe **ENGINE**

Metodi

- **carica_immagine(path: String, id_opera: Integer, num_pagina: Integer),** con questo metodo l'immagine viene caricata nel db tramite gli opportuni parametri;
- **cancella_immagine (id_immagine: Integer),** Cancella l'immagine relativa all'ID passato come parametro;
- **conferma_immagine (id_immagine: Integer),** mi imposta a TRUE l'attributo conferma;
- **crea_opera (nome: String),** Crea l'opera con il relativo titolo passato come parametro;
- **pubblica_opera(opera: OPERA),** rende visibile l'opera agli utenti;
- **modifica_opera (opera: OPERA),** modifica l'opera;
- **cancella_opera(id_opera: Integer),** elimina l'opera con il relativo ID passato come parametro;
- **carica_trascrizione(trascrizione: String, id_immagine: Integer),** carico la trascrizione relativa all'ID dell'immagine;
- **conferma_trascrizione (id_trascrizione: Integer),** mi imposta a TRUE l'attributo conferma;
- **modifica_trascrizione (trascrizione: TRASCRIZIONE),** mi sostituisce la trascrizione con quella passata come parametro;
- **cancella_trascrizione (id_trascrizione: Integer),** mi cancella la trascrizione che ha come id quello passa come parametro;
- **pubblica_trascrizione (id_trascrizione: Integer),** rende visibile la trascrizione agli utenti avanzati.

RELAZIONI TRA CLASSI

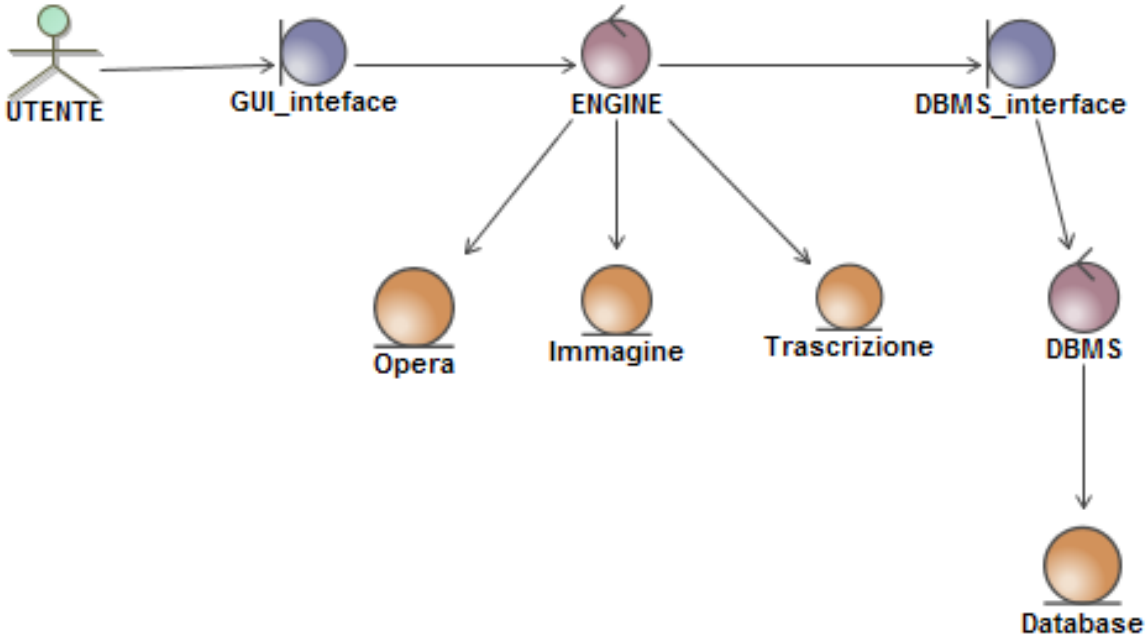
OPERA – IMMAGINE

Tra opera e immagine c'è una relazione 1 a N, perché un'opera può contenere una o più immagini, mentre un' immagine si riferisce ad una sola opera.
L'opera è inoltre composizione dell'immagine, ossia non può esistere alcuna opera se non c'è almeno un'immagine associata.

IMMAGINE – TRASCRIZIONE

Tra immagine e trascrizione c'è una relazione 1 a 1, perché ciascuna immagine fa riferimento ad una e una sola trascrizione e viceversa;
inoltre, la trascrizione è composizione dell'immagine, perché la trascrizione non può esistere se non esiste un' immagine.

ROBUSTNESS DIAGRAM



Per il nostro sistema abbiamo utilizzato il pattern architetturale MVC.

*Il **Model-View-Controller** (**MVC**, talvolta tradotto in italiano **Modello-Vista-Controllo**), in informatica, è un pattern architetturale molto diffuso nello sviluppo di sistemi software, in particolare nell'ambito della programmazione orientata agli oggetti, in grado di separare la logica di presentazione dei dati dalla logica di business.*

Il pattern è basato sulla separazione dei compiti fra i componenti software che interpretano tre ruoli principali:

- il **model** fornisce i metodi per accedere ai dati utili all'applicazione;
- il **view** visualizza i dati contenuti nel model e si occupa dell'interazione con utenti e agenti;
- il **controller** riceve i comandi dell'utente (in genere attraverso il view) e li attua modificando lo stato degli altri due componenti.