Progetto SIGEOL

Specifica Tecnica

v1.0.0

Redazione: Grosselle Alessandro, Scarpa Davide

26 maggio 2009

quixoft.sol@gmail.com

Verifica:	Barbiero Mattia
Approvazione:	Freo Matteo
Stato:	Formale
Uso:	Esterno
Distribuzione:	QuiXoft
	Rossi Francesca
	Vardanega Tullio
	Conte Renato

Sommario

Documento contenente la descrizione dettaglia dei componenti presenti nel sistema "SIGEOL" commissionato dalla prof. Rossi Francesca.

Indice

1 Introduzione

1.1 Scopo del documento

Il presente documento denominato Specifica tecnica ha lo scopo di mostrare la struttura del progetto *SIGEOL* e descrivere i componenti che ne fanno parte.

1.2 Scopo del prodotto

Il progetto sotto analisi, denominato *SIGEOL*, si prefigge di automatizzare la generazione, la gestione, l'ottimizzazione e la consultazione degli orari di lezione. Per maggiori dettagli consultare il documento denominato Analisi dei Requisiti alla sua ultima versione.

1.3 Glossario

Le definizioni dei termini specialistici usati nella stesura di questo e di tutti gli altri documenti possono essere trovate nel documento Glossario al fine di eliminare ogni ambiguità e di facilitare la comprensione dei temi trattati. Ogni termine la cui definizione è disponibile all'interno del Glossario verrà marcato con una sottolineatura.

2 Definizione del prodotto

2.1 Metodo e formalismo di specifica

Lo strumento principale nel redarre la specifica tecnica sarà il linguaggio UML, che permetterà di realizzare i diagrammi delle classi, di sequenza, di collaborazione e di attività.

La decomposizione architetturale utilizzata sarà di tipo Top-down. E' prevista una descrizione generale dell'architettura del sistema, alla quale seguiranno le specifiche dettagliate dei suoi componenti.

Per semplificare la progettazione, si utilizzeranno i seguenti pattern:

MVC Il pattern MVC (Model View Controller) si basa sulla separazione tra i componenti software del sistema, che gestiscono il modo in cui presentare i dati, e i componenti che gestiscono i dati stessi.

Façade Permette, attraverso un'interfaccia più semplice, l'accesso a sottosistemi aventi interfacce complesse e molto diverse tra loro, nonché a blocchi di codice complessi.

REST Representational state transfer (REST) è un tipo di architettura software per i sistemi di ipertesto distribuiti come il World Wide Web. REST si riferisce ad un insieme di principi di architetture di rete, i quali delineano come le risorse sono definite e indirizzate.

Convention Over Configuration Convention Over Configuration è un paradigma di programmazione che prevede configurazione minima (o addirittura assente) per il programmatore che utilizza un framework che lo rispetti, obbligandolo a configurare solo gli aspetti che si differenziano dalle implementazioni standard o che non rispettano particolari convenzioni di denominazione o simili. Significa che Rails prevede delle impostazioni di default per qualsiasi aspetto dell'applicazione. Utilizzando queste convenzioni sarà possibile velocizzare i tempi di sviluppo evitando di realizzare scomodi file di configurazione. L'esempio più chiaro del COC si può notare a livello di modelli: rispettando le convenzioni previste dal framework è possibile realizzare strutture di dati complesse con molte relazioni tra oggetti in pochissimo tempo, in maniera quasi meccanica e soprattutto senza definire nessuna configurazione. Questo concetto differenzia Rails dai framework che prevedono molte righe di configurazione per ogni aspetto dell'applicazione. Con il COC tutto diventa più snello e più dinamico. Ovviamente per situazioni in cui le convenzioni non possano essere rispettate, Rails permette di utilizzare schemi funzionali diversi da quelli previsti.

DRY Questo concetto, fortemente filosofico, prevede che ciascun elemento di un'applicazione debba essere implementato solamente una volta e niente debba essere ripetuto. Questo significa che, mediante Rails, è possibile gestire funzionalità ripetitive con una estrema fattorizzazione del codice ("scrivo una volta e uso più volte") che facilita sia lo sviluppo iniziale che eventuali modifiche successive del prodotto.

View Helper Questo pattern disaccoppia il Business Logic dallo strato View, il che facilita la manutenibilità. Aiuta a separare, in fase di sviluppo, la responsabilità del web designer e dello sviluppatore.

Active Record Secondo il pattern Active Record esiste una relazione molto stretta fra tabella e classe, colonne e attributi della classe.

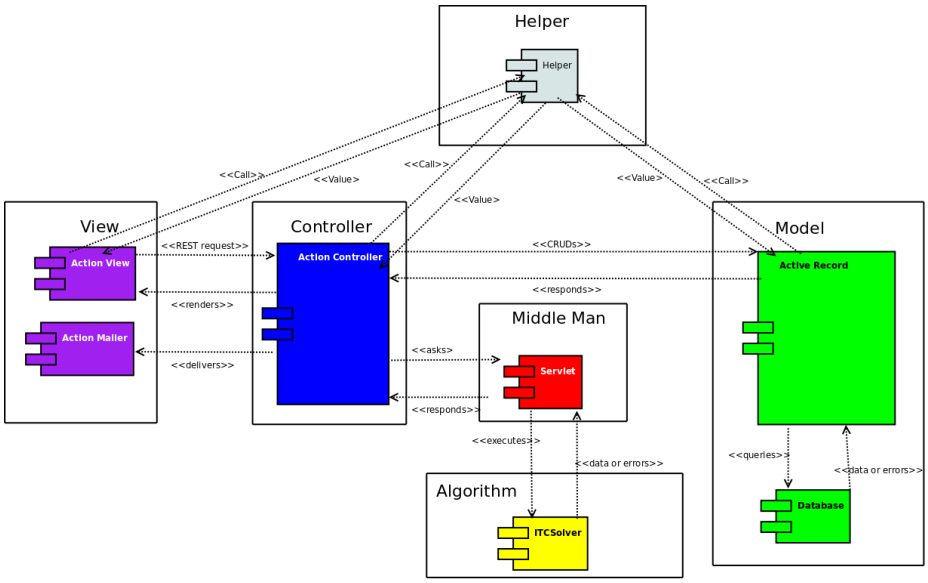
- una tabella di un database relazionale è gestita attraverso una classe
- una singola istanza della classe corrisponde ad una riga della tabella
- alla creazione di una nuova istanza viene creata una nuova riga all'interno della tabella, e modificando l'istanza la riga viene aggiornata

Il sistema verrà implementato utilizzando Ruby on Rails, un framework la cui architettura è fortemente ispirata al paradigma Model-View-Controller. Oltre a veicolare lo sviluppo di applicazioni secondo il pattern MVC, Rails segui la filosofia REST imponendo un'ulteriore disciplina nella codifica degli indirizzi garantendo quindi maggior chiarezza e compattezza.

2.2 Presentazione dell'architettura generale del sistema

Per presentare l'architettura generale del sistema *Sigeol* si utilizzerà il pattern Model-View-Controller(MVC). In questo modello i ruoli di presentazione, controllo ed accesso ai dati vengono affidati a componenti diversi e sono tra di loro disaccoppiati.

Sono inoltre previste altre due componenti: Middle Man si occupa di ricevere i compiti che necessitano di un elevato carico di lavoro; Algorithm, invece, risolve il problema del calcolo dell'orario.



3 Descrizione dei singoli componenti

3.1 Model

L' ActiveRecord è il modulo di Ruby on Rails che gestisce la persistenza dei dati. Il modulo è stato implementato seguendo il pattern Active Record facendo largo uso di convenzioni sui nomi da assegnare a tabelle, classi, colonne e attributi; se non sono presenti configurazioni particolari del modello, sono valide le seguenti convenzioni:

- per ogni modello esiste una tabella avente come nome il nome del modello al plurale e in caratteri minuscoli;
- nel caso di modelli con nome composto di più parole, il nome del modello ha la prima lettera di ogni parola; maiuscola, mentre la tabella ha sempre il nome tutto in minuscolo e separa le parole con un trattino basso (underscore);

- per ogni colonna della tabella viene reso disponibile il relativo attributo con lo stesso nome;
- ogni tabella ha una colonna id (intero positivo) che identifica univocamente ogni record;

La classe è descritta in buona parte dalla tabella che rappresenta ed espone i metodi necessari per leggere e scrivere i record. Con ActiveRecord è possibile utilizzare diversi tipi di database: SQLite, MySQL, Postgresql, Oracle, IBM DB/2 ed è possibile scrivere driver personalizzati per altri database relazionali.

3.1.1 Relazione d'uso con le altre componenti

Questa componente si relaziona con il Controller fornendo i dati che vengono richiesti e salvando le eventuali modifiche al database.

3.2 Controller

Il modulo che gestisce la parte controller è contenuto all'interno del framework Ruby on Rails ed è chiamato Action Controller. Questo, assieme all'Action View(confronta sezione 3.3) è integrato all'interno di un pacchetto chiamato Action Pack.

Ogni controller è una normale classe, ed ogni metodo pubblico definito in questa classe corrisponde ad un'azione specifica.

Ruby on Rails riconosce il tipo di richiesta pervenuta codificando le informazioni all'interno di un URL e si serve di un componente chiamato Rounting, per determinare l'azione cui deve essere sottoposta la richiesta. Il componente Routing traccia una mappatura che permette a Rails di collegare gli URL esterni e l'azione contenuta in una determinata classe Controller.

Ad esempio, dato un URL nel formato *controller/azione/id*, viene identificato il Controller *controller* e viene istanziato. A questo punto l'oggetto richiama il metodo con nome *azione* e con parametro *id*. Il Controller infine cercherà di visualizzare un template con lo stesso nome dell'azione.

3.2.1 Relazione d'uso con le altre componenti

Questa componente si relazione con la View, il Model ed il Middle Man.

Il Controller funge da intermediario fornendo i dati alla View attraverso il Model. Inoltre sceglie quale view renderizzare a seconda della richiesta pervenuta.

3.3 View

Il modulo Action View contenuto nel framework Ruby on Rails, offre meccanismi avanzati per il riutilizzo del codice, tramite l'uso di viste, layouts, partials e di metodi helper pensati per generare ad esempio pagine XHTML. I metodi helper sono semplici metodi pubblici di una classe Controller.

3.3.1 Relazione d'uso con le altre componenti

Questa componente si trova in relazione solamente con il Controller. Converte infatti le azioni che l'utente effettua attraverso la View in richieste a metodi propri del Controller.

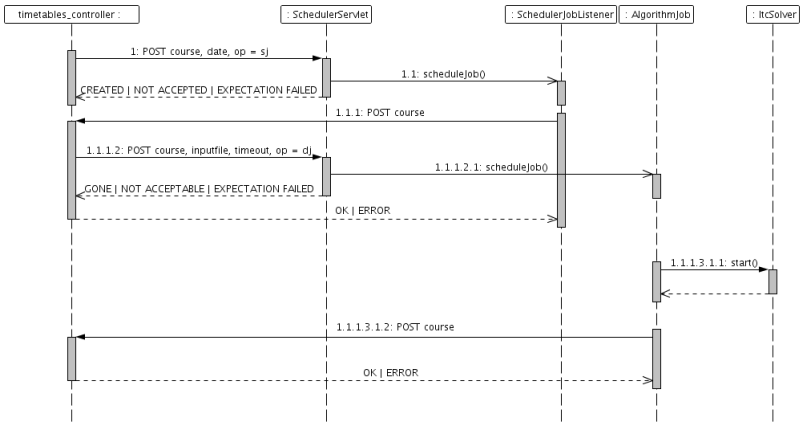
3.4 Middle Man

La componente Middle Man è estranea al pattern MVC e si occupa di gestire quelle richieste che a causa del tempo di calcolo, renderebbero inutilizzabile ogni altra funzione di consultazione dell'applicazione web. Questa componente sarà implementata tramite una servlet in Java denominata SchedulerServlet. La componente integrerà Quartz (un' open source job scheduling system, per maggiori informazioni visitare http://www.opensymphony.com/quartz/) il quale verrà utilizzato per gestire tutte le operazioni di scheduling e fornirà strumenti aggiuntivi (quali ripristino informazioni dopo crash di sistema e gestione schedulazioni in formato cron).

3.4.1 Relazione d'uso con le altre componenti

Questa componente è in relazione con il Controller e l'Algorithm.

Per soddifare le richieste del Controller preleva i dati necessari da un file, creato dall'applicazione stessa, e si occuperà poi di avviare la componente Algorithm per il calcolo degli orari passandogli tutti i dati necessari. Inoltre la componente è utilizzata dal Controller per l'inserimento delle tempistiche di esecuzione del calcolo degli orari.



- 1: impostazione di una nuova generazione dell'orario di un corso di laurea ad una specifica data
- 1.1: impostazione della classe da istanziare all'avvio dell'evento schedulato
- 1.1.1: invio della notifica all'applicazione per la preparazione file di input e avvio calcolo
- 1.1.1.2: invio dei dati alla servlet utili alla generazione dell'orario (manuale e schedulata)
- 1.1.1.2.1: impostazione dell'istanza di classe che si occuperà di avviare l'algoritmo
- 1.1.1.3.1.1: avvio dell'algoritmo
- 1.1.1.3.1.2: segnalazione della generazione dell'orario effettuata (o di eventuale errore)

3.5 Algorithm

La componente Algorithm non inclusa nel pattern MVC si occupa di calcolare effettivamente l'orario delle lezioni, rispettando i vincoli inseriti nel sistema. Dagli studi di ricerca operativa sono noti diversi algoritmi per la ricerca di una soluzione ottimale di un problema sottoposto a vincoli. L'algoritmo scelto è stato utilizzato nell'International Timetabling Competition 2007 ed è basato sulla libreria Constraint Solver Library (licenza GNU LGPL).

3.6 Relazioni d'uso di altre componenti

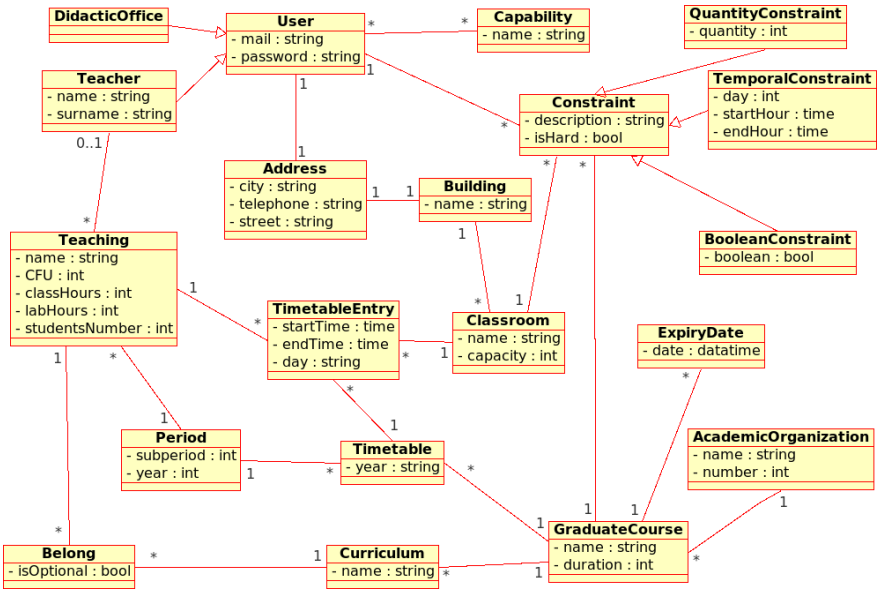
Questa componente si trova in relazione solo con la componente Middle Man. Algorithm restituisce il risultato in un file che verrà poi processato dall'applicazione.

4 Diagrammi delle classi

4.1 Model

Rails preferisce le convenzioni alle configurazioni, e cerca di evitare allo sviluppatore il peso di dover specificare l'associazione tra tabelle e classi. Non è quindi necessario illustrare nella specifica tecnica la struttura del database, in quanto verrà implementato da Rails a partire dalla struttura delle classi.

L'utilizzo dell'inglese nella scelta dei nomi delle classi è necessario per rispettare le convenzioni di Rails. I nomi delle tabelle del database di sistema saranno semplicemente il plurale dei nomi delle classi qui di seguito specificate:



- **Address**: rappresenta la classe che individua un indirizzo civico.
- **Belong**: rappresenta la classe che individua se un determinato insegnamento, inserito in un preciso corso di laurea, e' opzionale o meno.
- **Building**: rappresenta la classe che gestisce i dati di un complesso. Quest'ultimo possiede aule, laboratori e uffici dei docenti.

- **Capability**: rappresenta la classe che gestisce le informazioni di un determinato servizio, disponibile per gli utenti del sistema. Ogni tipologia di utente ha uno specifico numero di servizi a sua disposizione.

- **Constraint**: rappresenta la classe che individua un vincolo o una preferenza, necessario/a al calcolo dello schema d'orario. L'attributo booleano isHard indica se è un vincolo o una preferenza.

  QuantityConstraint: rappresenta la classe che individua un vincolo di tipo numerico; ad esempio il numero massimo di ore di una lezione. La classe QuantityConstraint è in associazione polimorfa con la classe Constraint.

  TemporalConstraint: rappresenta la classe che individua un vincolo di tipo temporale, specificato da un giorno della settimana, un'ora di inizio ed un'ora di fine. La classe TemporalConstraint è in associazione polimorfa con la classe Constraint.

  BooleanConstraint: rappresenta la classe che individua un vincolo che potrà essere considerato o meno dall'algoritmo in base al valore del suo attributo booleano. La classe BooleanConstraint è in associazione polimorfa con la classe Constraint.

- **Curriculum**: rappresenta la classe che individua un percorso formativo appartenetente ad un corso di laurea.

- **ExpiryDate**: rappresenta la classe che individua la data oltre la quale un docente non potrà più inserire e/o modificare i propri vincoli e preferenze. Successivamente a questa data l'algoritmo di calcolo dello schema d'orario si avvierà.

- **GraduateCourse**: rappresenta la classe che individua un corso di laurea e ne gestisce i relativi dati.

- **AcademicOrganization**: rappresenta la classe che descrive com'è strutturato un corso di laurea: bimestre, trimestre, semestre.

- **Period**: rappresenta la classe che individua una fascia temporale. Le fascie temporali sono generate in base alla durata e alla struttura del corso di laurea (bimestre, trimestre, semestre). Ogni insegnamento è inserito in una specifica fascia temporale; ad esempio primo (year) anno, primo (subperiod) trimestre.

- **Teaching**: rappresenta la classe che gestisce i dati relativi ad un determinato insegnamento.

- **Timetable**: rappresenta la classe che permette di individuare lo schema d'orario specifico di un corso di laurea e di un periodo (bimestre, trimestre, semestre)

- **TimetableEntry**: rappresenta la classe che individua un giorno, un'ora di inizio e un'ora di fine di un determinato insegnamento.
- **User**: rappresenta la classe che gestisce i dati di login. In particolare possiede i seguenti due attributi: mail (che funge da username) e password (che permette l'autenticazione sicura).

  DidacticOffice: rappresenta la classe che gestisce i dati della segreteria. Questa classe e' in associazione polimorfa con la classe User

  Teacher: rappresenta la classe che gestisce i dati personali di un docente. Questa classe e' in associazione polimorfa con la classe User.

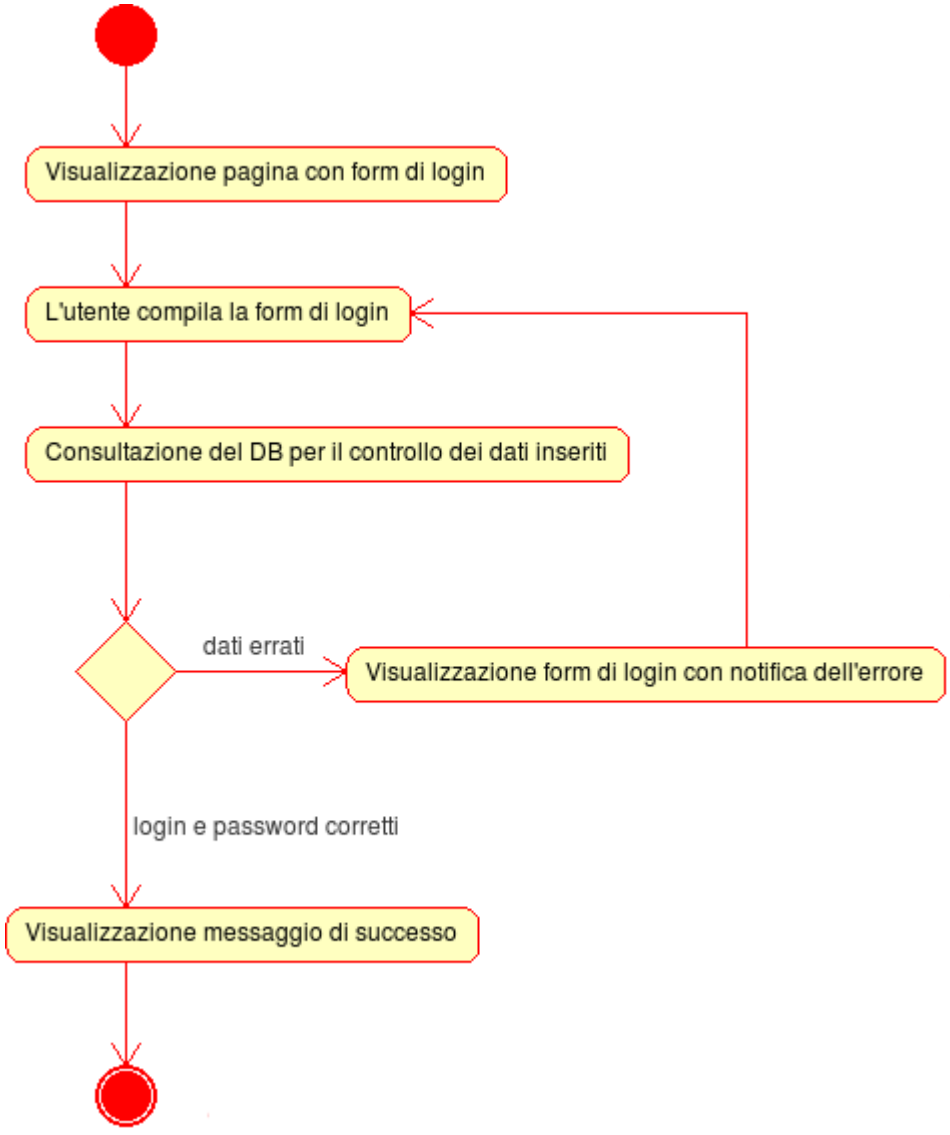
5 Flusso di esecuzione

La seguente sezione si fa carico di illustrare, con l'utilizzo di diagrammi di attività, i flussi delle varie azioni che gli utenti del sistema SIGEOL hanno la possibilità di compiere. Quanto riportato in seguito è da considerarsi passibile di modifiche col procedere della fase di progettazione.

5.1 Login utenti

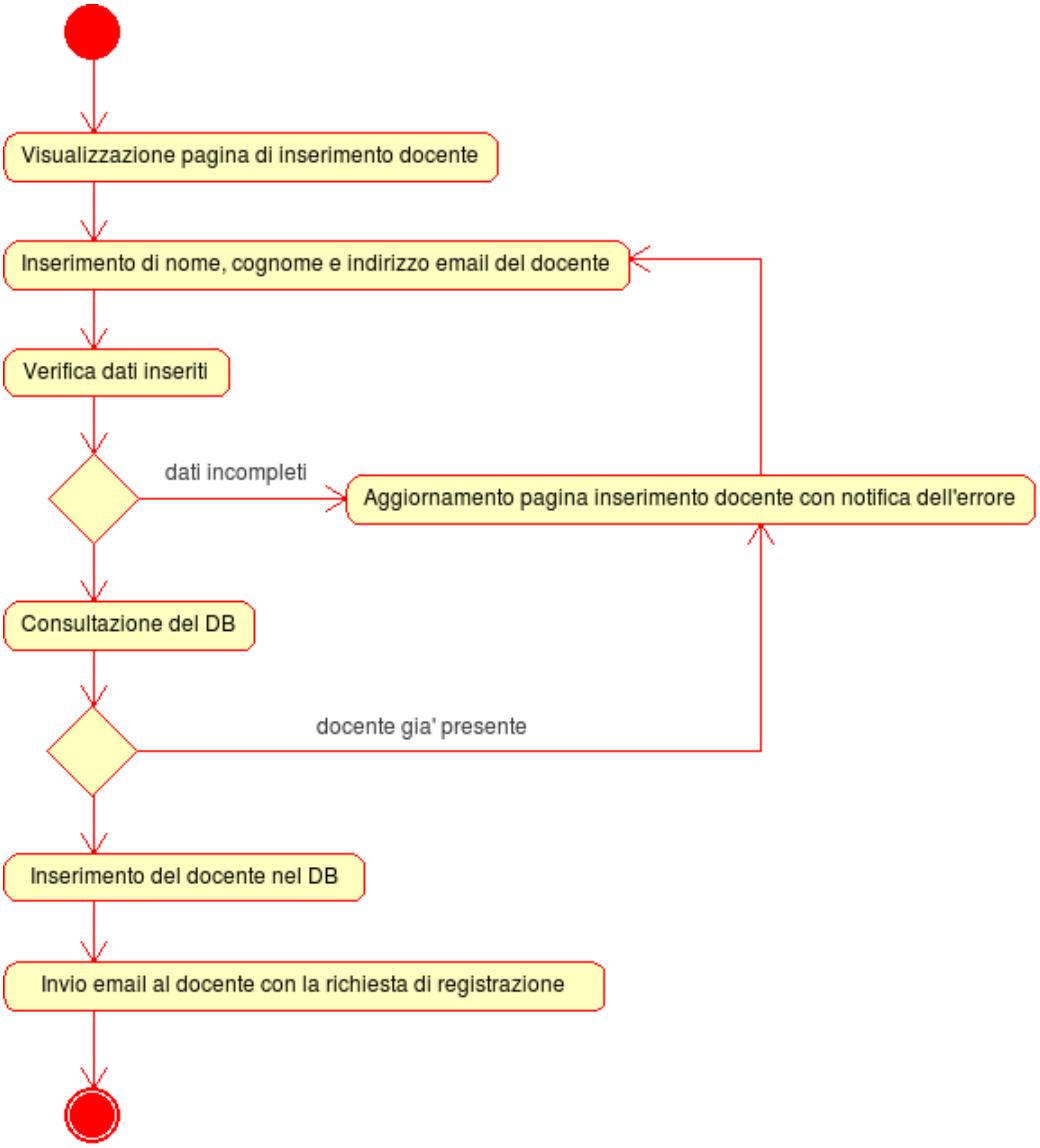
La Segreteria Didattica e tutti Docenti registrati hanno la possibilità di accedere alla pagina di login per inserire le proprie credenziali ed accedere alle funzioni aggiuntive che il sistema SIGEOL mette loro a disposizione. Il Presidente del CCS viene considerato un Docente, ma con in più la possibilità di accedere a tutte le funzioni tipiche della Segreteria Didattica.

Al termine della fase di login, l'utente verrà reindirizzato alla pagina principale di SIGEOL e potrà espletare le funzioni a sua disposizione.



5.2 Inserimento nuovo Docente

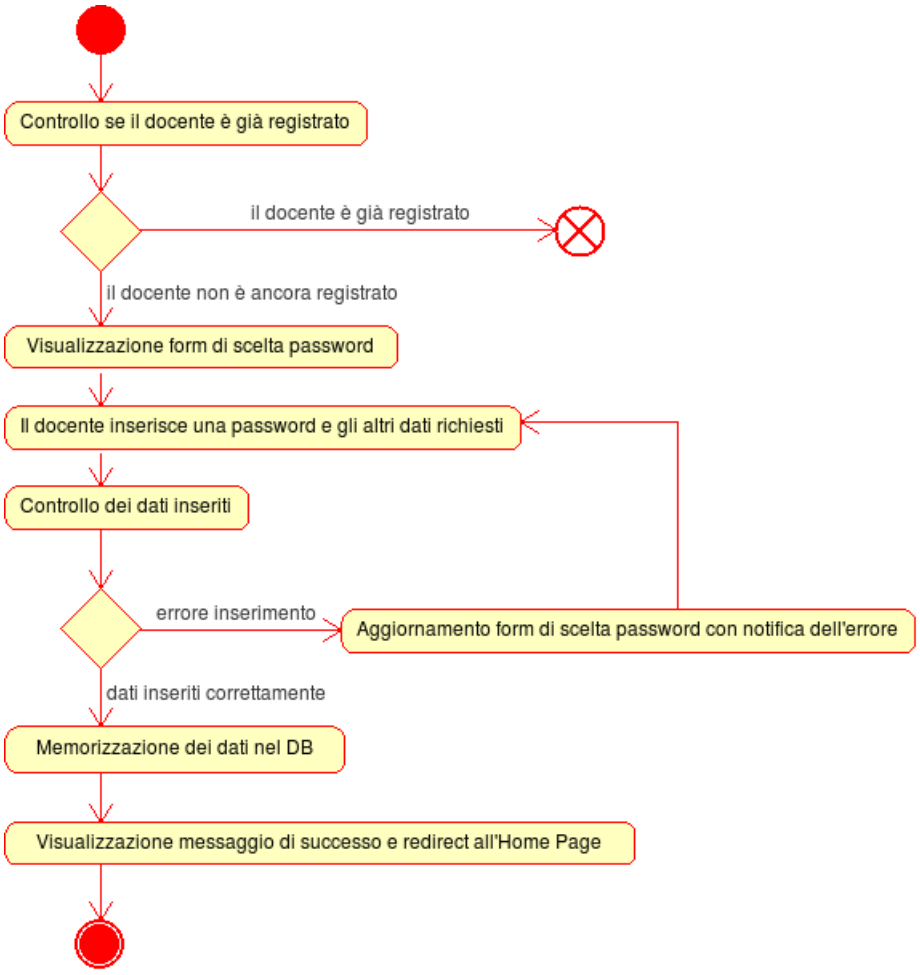
La Segreteria Didattica e il Presidente del CCS hanno la possibilità, dopo aver effettuato con successo il login, di inserire un nuovo Docente nel sistema SIGEOL. Verranno richiesti nome, cognome ed indirizzo email. Se tali dati vengono inseriti e il Docente non è già presente nel DB di sistema, viene automaticamente inviata una email al Docente invitato, in cui gli viene indicato il link necessario per procedere alla creazione del suo account.



5.3 Registrazione Docente

Ogni docente, dopo aver ricevuto la mail di invito al sistema SIGEOL, utilizza il link ricevuto per accedere al form di scelta password e di inserimento degli altri dati personali. La scelta della password seguirà rigide regole di sintassi, che verranno controllate dinamicamente durante l'inserimento. Tali regole prevederanno un numero minimo di caratteri, la presenza obbligatoria di cifre o simboli e altre convenzioni che saranno decise dal team QuiXoft, in collaborazione con il Committente, durante il proseguimento della fase di progettazione. La password dovrà poi essere confermata in un successivo campo dati, per evitare spiacevoli errori di battitura.

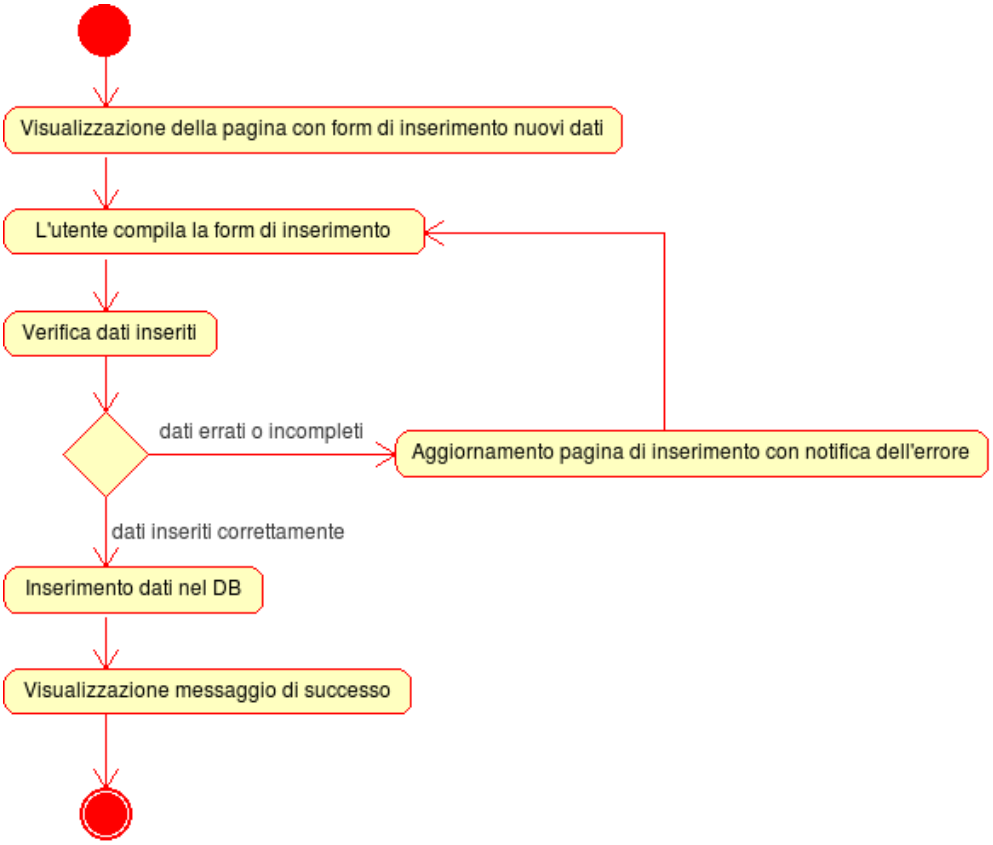
Al termine della prcedura la password scelta e gli altri dati inseriti verranno memorizzati in modo sicuro sul DB di sistema.



5.4 Inserimento dei dati nel sistema

La Segreteria Didattica e il Presidente del CCS hanno la possibilità di inserire i dati relativi ai corsi di laurea, agli insegnamenti, alle aule, ai dipartimenti e molti altri dati indispensabili al funzionamento del sistema SIGEOL.

L'inserimento avverrà tramite diverse pagine web e diverse form, ma il diagramma di flusso di tali inserimenti è cosi simile che è stato scelto di esporne solo uno generale, adatto a rappresentare tutti gli inserimenti.

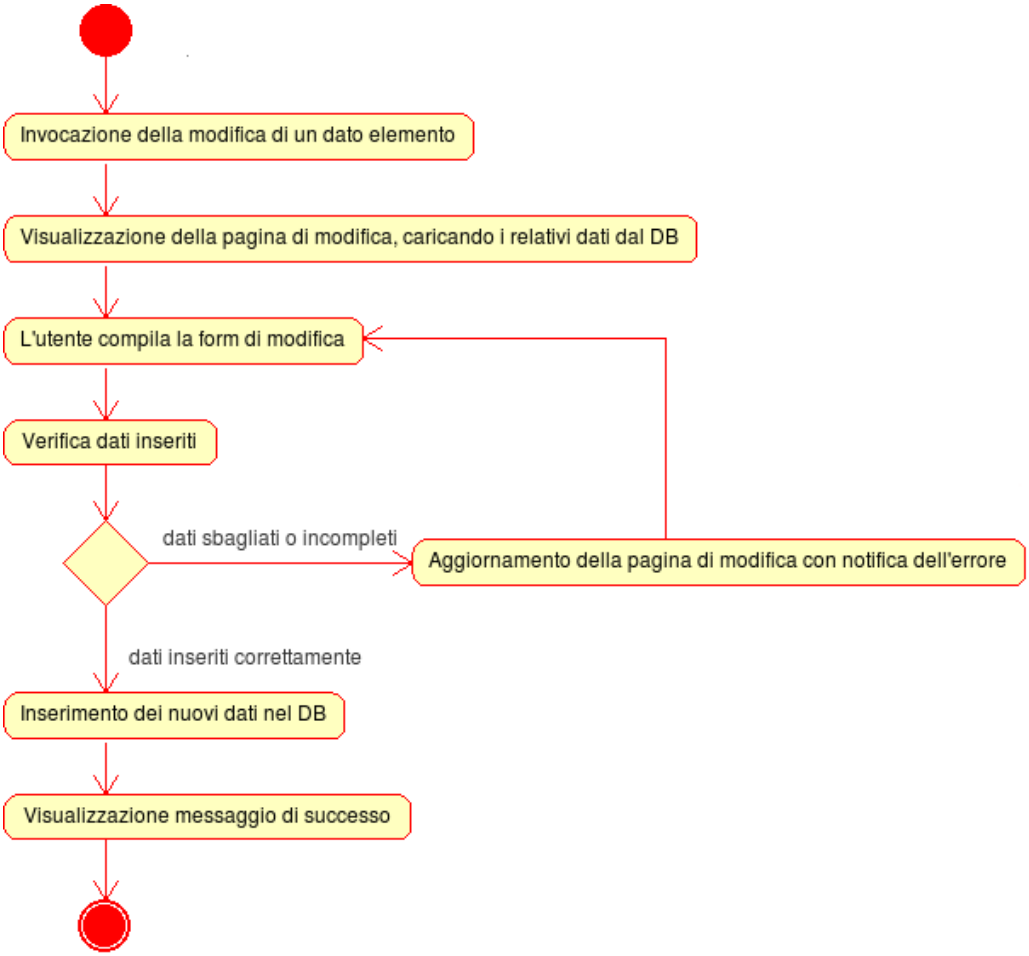


5.5 Modifica dei dati del sistema

La Segreteria Didattica e il Presidente del CCS hanno la possibilità di modificare tutti i dati relativi ai corsi di laurea, agli insegnamenti, alle aule, ai dipartimenti, ecc...

I docenti invece hanno la possibilità di modificare i propri dati personali e i vincoli e le preferenze inserite.

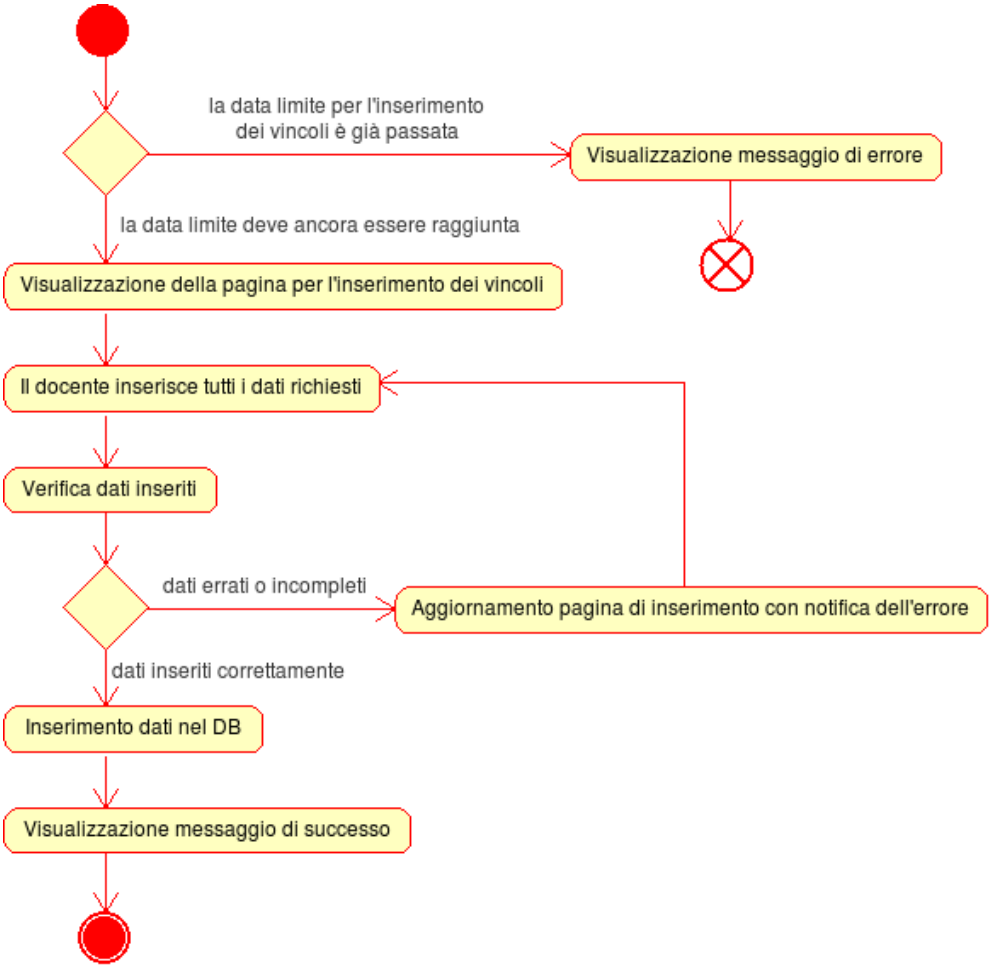
Le similitudini tra tutte le sopra citate modifiche hanno portato allo sviluppo di un singolo diagramma di attività che le rappresenti tutte. L'invocazione del comando di modifica sarà ovviamente diverso tra i vari tipi di dati, ma la procedura per modificarli e salvarli è similare.



5.6 Inserimento vincoli e preferenze da parte dei Docenti

Tutti i docenti cha abbiano effettuato correttamente il login hanno la possibilità di inserire i loro vincoli e le loro preferenze. La procedura è similare, per tanto è rappresentata dallo stesso diagramma di attività.

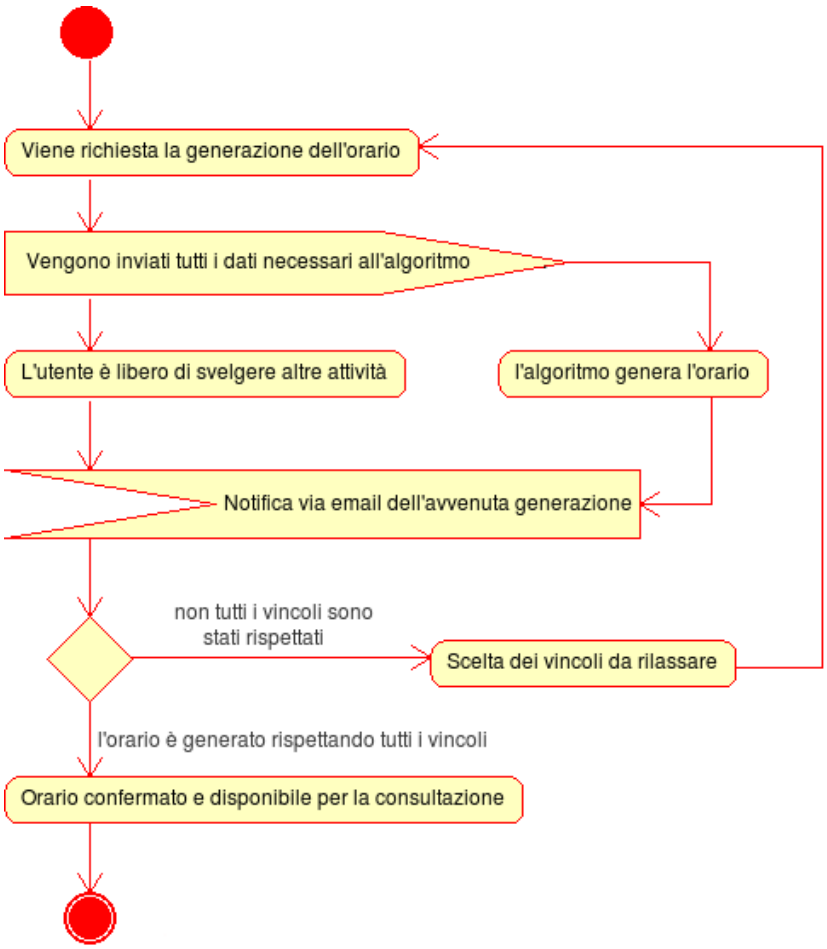
La differenza sostanziale nell'inserimento di un vincolo o di una preferenza è che solo il vincolo, tra i due, prevede una motivazione, mentre solo la preferenza prevede un livello di priorità. L'assegnazione della priorità di una preferenza è automatizzata dal sistema: il Docente dovrà solamente decidere l'ordine di importanza delle proprie preferenze.



5.7 Generazione dell'orario

La Segreteria Didattica e il Presidente del CCS hanno la possibilità, una volta scaduta la data limite per l'inserimento dei vincoli e delle preferenze da parte dei Docenti, di far partire l'algoritmo di generazione dell'orario. L'algoritmo calcolerà l'orario migliore possibile per tutti i Corsi di Laurea inseriti, e sarà data la possibilità di scelta se generare l'orario per uno o più periodi (per periodo si intende il numero di trimestre-semestre-ecc... in base alla scelta fatta nel momento di inserimento dei dati).

L'algoritmo proporrà un orario soddisfacendo tutti i requisiti presenti e tentando di rispettare il più possibile tutte le preferenze. Se ciò non fosse possibile, sarà data la possibilità alla Segreteria Didattica o al Presidente del CCS di rilassare certi vincoli o di modificare manualmente lo schema d'orario generato.

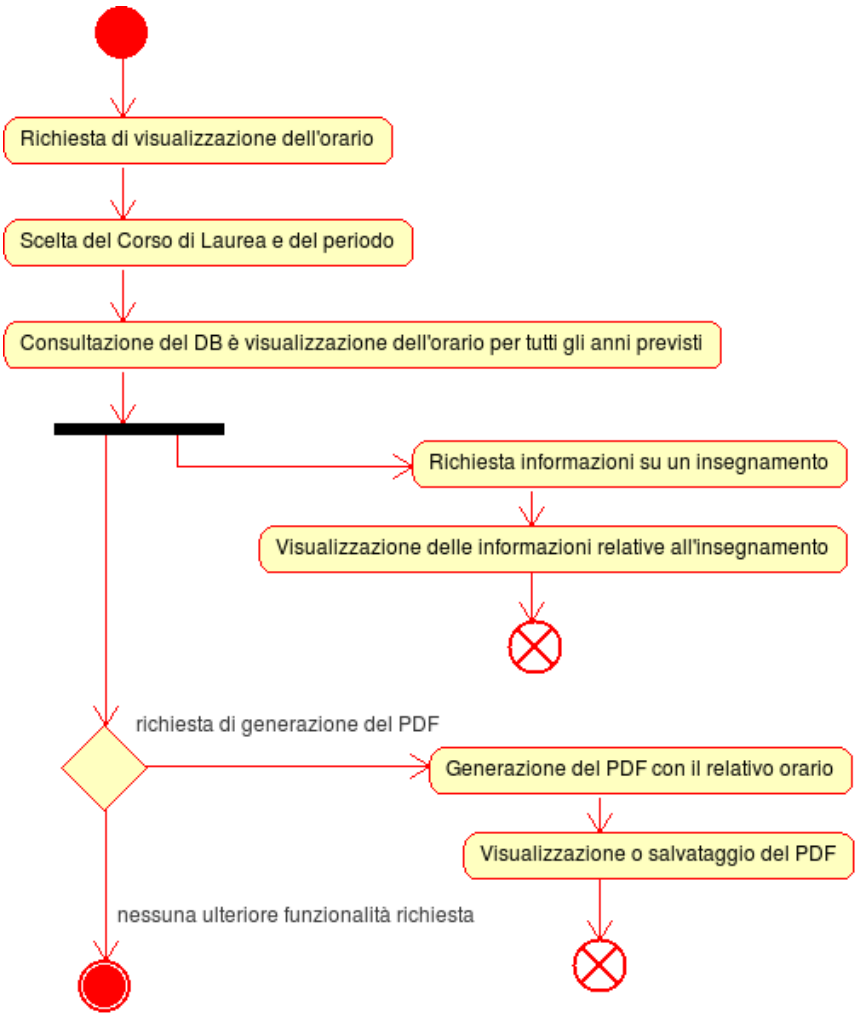


5.8 Consultazione dell'orario

Qualsiasi utente ha la possibilità di consultare gli orari generati dal sistema SIGEOL, anche senza effettuare alcuna operazione di login. Sarà sufficiente scegliere il Corso di Laurea ed il periodo e verrà visualizzato il relativo orario.

Cliccando su un determinato insegnamento sarà possibile consultarne le informazioni. Quest'operazione può essere compiuta quante volte si vuole, senza modificare la visualizzazione dell'orario: le informazioni saranno infatti mostrate su una nuova pagina o su una finestra pop-up. Con il proseguire della fase di progettazione verrà definitivamente decisa la forma di visualizzazione.

In ogni momento è data la possibilità all'utente di generare un PDF contenente l'orario visualizzato: sarà possibili visualizzarlo all'interno del proprio browser oppure salvarlo.



6 Tracciamento componenti-requisiti

Di seguito è riportato il tracciamento dei requisiti descritti nel documento denominato Analisi dei Requisiti con le componenti descritti nel presente.

Requisiti	View	Controller	Model	MiddleMan	Algorithm
RFO0.0	✓	✓	✓		
RFO0.1	✓	✓	✓		
RFO0.2	✓	✓	✓		
RFO0.3	✓	✓	✓		
RFO0.4	✓	✓	✓		
RFO0.5	✓	✓	✓		
RFO0.6	✓	✓	✓		
RFO0.7	✓	✓	✓	✓	✓
RFO0.8	✓	✓	✓		
RFO0.9	✓	✓	✓		
RFO0.10	✓	✓	✓		
RFO0.11	✓	✓	✓		
RFO0.12	✓	✓	✓		
RFO0.13	✓	✓	✓		
RFO0.14	✓	✓	✓		
RFO0.15	✓	✓	✓		
RFO0.16	✓	✓	✓		
RFO0.17	✓	✓	✓		
RFO0.18	✓	✓	✓		
RFO0.19	✓	✓	✓		
RFO0.20	✓	✓	✓		
RFO0.21	✓	✓	✓		
RFO0.22	✓	✓	✓		
RFO0.23	✓	✓	✓		
RFO0.24	✓	✓	✓		

Requisiti	View	Controller	Model	MiddleMan	Algorithm
RFO1.0	✓	✓	✓		
RFO1.1	✓	✓	✓		
RFO1.2	✓	✓	✓		
RFO1.3	✓	✓	✓		
RFO1.4	✓	✓	✓		
RFO1.5	✓	✓	✓		
RFO1.6	✓	✓	✓		
RFD0.0	✓	✓	✓		
RFD0.1	✓	✓	✓		
RFD0.2			✓	✓	
RFD0.3	✓	✓	✓		
RFD1.0	✓	✓	✓		
RFD2.0	✓	✓	✓		
RFD2.1	✓	✓	✓		
RQO0	✓				
RQO1			✓		
RQO2		✓	✓		
RQO3	✓	✓	✓	✓	✓
RQO4	✓				
RQO5	✓				
RQO6	✓	✓	✓	✓	✓
RQD1		✓	✓		
RIO0.0			✓		
RIO1.0	✓	✓	✓		
RIO1.1	✓	✓	✓		
RIO1.2	✓				
RID0.0	✓	✓	✓	✓	✓

Diario delle modifiche

Data	Versione	Modifica
07-03-2009	1.0.0	Approvazione del responsabile e passaggio di stato a Formale
11-02-2009	0.7.0	Aggiunta diagramma di sequenza del Middle Man
10-02-2009	0.6.4	Riscrittura contenuto Middle Man
24-01-2009	0.6.3	Sottolineatura dei termini del Glossario
24-01-2009	0.6.2	Correzioni di forma e sintassi
24-01-2009	0.6.1	Lieve correzione nel diagramma delle classi
23-01-2009	0.6.0	Aggiunto il tracciamento requisiti-componenti
22-01-2009	0.5.0	Aggiunta il diagramma e la descrizione delle classi
21-01-2009	0.4.1	Correzioni varie di tipo ortografico
21-01-2009	0.4.0	Aggiunto middleman e algorithm
20-01-2009	0.3.1	Correzioni varie
20-01-2009	0.3.0	Aggiunti i flussi di esecuzione
17-01-2009	0.2.0	Stesura Descrizione dei componenti
15-01-2009	0.1.0	Creazione dell'indice