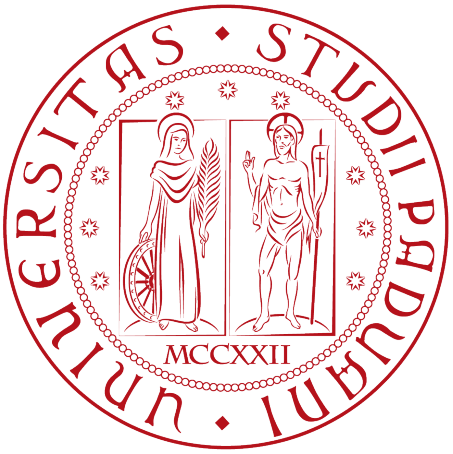
Università degli Studi di Padova

Dipartimento di Matematica "Tullio Levi-Civita"

Corso di Laurea in Informatica



Progettazione e implementazione di una soluzione BI per la gestione di processi di budget

Tesi di laurea triennale

Relatore
Prof. Bujari Armir

Laureando
Tomas Mali

Anno Accademico 2017-2018

Tomas Mali: *Progettazione e implementazione di una soluzione BI per la gestione di processi di budget ,* Tesi di laurea triennale, © Aprile 2018.

Sommario

Il presente documento descrive il lavoro svolto durante il periodo di stage, della durata di circa trecento ore, dal laureando Tomas Mali presso l'azienda Sanmarco Informatica S.p.A.

Gli scopi sono stati la progettazione e la successiva implementazione di una soluzione BI (business intelligence) efficace ed efficiente per gestire un insieme di processi aziendali che riguardano la raccolta e l'analisi dati in modo rapido e sicuro, minimizzando i futuri costi di manutenzione dell'applicativo preesistente.

Gli obiettivi principali sono stati quindi lo studio e la valutazione delle tecnologie innovative all'avanguardia per l'elaborazione e la manipolazione di dati gestioneali di grandi dimensioni, al fine di renderli di immediata disponibilità e portabilità, presentandoli, per quanto riguarda l'interazione con l'utente, attraverso una semplice e dinamica interfaccia mobile, affiancata da un'interfaccia web preesistente. Trattandosi dunque di raccolte dati provenienti da una fonte gestionale, risulta evvidenta quindi l'attenzione e l'accuratezza che bisogna prestare nel manipolare tali dati e presentarli, ad esempio sotto forma di documenti potatili come PDF, txt eccetera.
In secondo luogo inoltre è stato richiesto un miglioramento delle funzionalità dell'applicativo preesistente, in particolare un miglioramento nella gestione degli utenti (di vari ruoli come capoarea, superuser ecc..) i quali usano in modo profilato i processi a seconda del tipo di utente.

"Life is really simple, but we insist on making it complicated"

— Confucius

Ringraziamenti

Innanzitutto, vorrei esprimere la mia gratitudine al Prof. Bujari Armir, relatore della mia tesi, per l'aiuto e il sostegno fornitomi durante la stesura del lavoro.

Desidero ringraziare con affetto i miei genitori per il sostegno, il grande aiuto e per essermi stati vicini in ogni momento durante gli anni di studio.

Ho desiderio di ringraziare poi i miei amici per tutti i bellissimi anni passati insieme e le mille avventure vissute.

Padova, Aprile 2018 Tomas Mali

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

In questo capitolo viene presentata brevemente l'azienda Sanmarco Informatica S.p.A. presso cui è stato svolto lo stage e la necessità che ha fatto nascere l'idea di questo stage.
Inoltre si presentano la struttura dei capitoli della tesi ed alcune norme tipografiche che verranno usate all'interno della stessa.

1.1 L'azienda

Sanmarco Informatica nasce negli anni '80 come *Software house*$_G$ specializzata nello sviluppo di applicazioni gestionali per aziende manifatturiere ed è oggi una *leading company* italiana nella progettazione e realizzazione di soluzioni a supporto della riorganizzazione di vari processi aziendali e professionali. L'ambizione e la volontà di rinnovarsi hanno permesso all'azienda di evolversi attraverso esperienze e scelte imprenditoriali di successo, che individuano nella specializzazione del proprio capitale umano l'elemento centrale. L'azienda, partner di *IBM Italia*$_G$, cresce grazie all'impegno di 320 persone fra dipendenti e collaboratori, 13 distributori e 4 sedi: Grisignano di Zocco (VI) come sede principale e Reggio Emilia (RE), Tavagnacco (UD) e Vimercate (MB) come filiali. Sanmarco Informatica è la prima ed unica azienda italiana entrata a far parte dell'*Open Power Foundation IBM*$_G$ e a gennaio 2016 ha ricevuto il riconoscimento internazionale *Beacon Award* come finalisti a livello mondiale fra le aziende d'eccellenza che propongono soluzioni tecnologiche innovative in combinazione con il sistema *Power*$_G$ di IBM.

1.2 L'idea

L'idea di base del progetto di stage si basa sulla necessità di alcune aziende di gestire in maniera più efficiente ed immediata i loro ordini giornalieri, la disponibilità degli articoli, gli scadenzari, gli incassi ed altri processi. Questo acquisisce ancora maggiore importanza laddove l'azienda in questione disponga di un numero elevato di rappresentanti ed ognuno di questi dovrà gestire i punti descriti sopra in base al proprio ruolo aziendale.

NextBI, che è uno dei team dell'azienda Sanmarco Informatica S.pA, offre attualmente una soluzione (che si sta espandendo con altre funzionalità) a questo problema, interrogando il database gestionale e sucessivamente impaginando il risultato su una pagina web. Nasce così la neccessità di affiancare l'applicativo web con un sistema di BI, il quale dovrà interaggiare con il front-end, rappresentabile mediante un'interfaccia mobile portabile, dinamica e di facile intuizione. L'ide nasce quindi anche dall'opportunità di poter gestire l'abilitazione dell'applicativo con le loro licenze, agli utenti desiderati in modo rapido senza effettuare tante operazioni. Nel automatizzare questa funzionalità sorge la necessità di studiare una soluzione nella quale tutta la parte di autenticazione per categoria di utenti venga gestita in maniera imediata, più dinamica e rapida senza il bisogno quindi di accedere a pagine web, ovviando cosi la necessità di autenticazione ripetuta.

1.3 Organizzazione del testo

Il secondo capitolo descrive con maggiore attenzione i dettagli dello stage. Specifica con più precisione la necessità a cui rispondere ai requisiti e gli obiettivi previsti, le tecnologie usate e la pianificazione prevista;

Il terzo capitolo descrive in breve i motivi ed alcuni problemi legati alla necessitaà di un sistema portabile, della trasformazione dei dati e delle notifiche in tempo reale.

Il quarto capitolo descrive in maggiore dettaglio il lavoro eseguito durante lo stage, le varie soluzioni che sono state testate e come queste sono state scelte e composte per progettare la soluzione ideale.

In questo capitolo si descrivono le principali scelte tecnologiche, la struttura del database,la struttura server-side dell'applicativo, il modello Long Polling implementato da Telegram la struttura client-side dell'applicativo.

Questo è il capitolo conclusivo dove vengono descritte le relazioni finali dell'esperienza dello stage presso l'azienda Sanmarco Informatica S.p.A. Sarà mostrato un riassunto generale che comprende gli obiettivi del progetto di stage, una descrizione delle conoscenze acquisite durante questo perido ed infine una valutazione personale riguardo l'intera esperienza di stage presso l'azienda.

Riguardo la stesura del testo, relativamente al documento sono state adottate le seguenti convenzioni tipografiche:

* gli acronimi, le abbreviazioni e i termini ambigui o di uso non comune menzionati vengono definiti nel glossario, situato alla fine del presente documento;
* per la prima occorrenza dei termini riportati nel glossario viene utilizzata la seguente nomenclatura: *parola*[g];
* i termini in lingua straniera o facenti parti del gergo tecnico sono evidenziati con il carattere *corsivo*.

Capitolo 2

Descrizione dello stage

In questo capitolo viene descritto il modo più dettagliato il progetto dello stage, presentati gli obiettivi e le pianificazioni previste. infine vengono illustrate le tecnologie usate per la realizzazione del progetto.

2.1 Il progetto di stage

L'azienda Sanmarco Informatica S.p.A. fornisce un servizio di BI per la gestione dei processi di budget. La soluzione attuale si basa su un applicativo web B2B. Essendo tale applicativo realizzato con tecnologie poco recenti si è mirato quindi a migliorarlo, aggiornando le tecnologie, mantenendo però la gran parte della struttura esistente invariata. A questo scopo il team NextBI si è preso l'impegnativa di estenderlo, migliornado le funzionalità esistenti e aggiungendo delle altre nuove, dando il nome al progetto, appunto, Budget. In base alla pianificazione, la dimensione di questo progetto risulta molto grande (stimato il rilascio nell'ottobre 2018). Da questa stima si deduce infatti che l'interesse del progetto non si limita solo all'aggiornamento della versione dell'applicativo esistente, ma bensì integrando man mano nel tempo migliorie e servizi innovativi. Il mio progetto si inserisce quindi all'interno del progetto budget con lo scopo di realizzare un sistema portabile e flessibile per la gestione e il monitoraggio di alcuni processi aziendali come sono ad esempio: gli ordini degli articoli giornalieri (settimanali o mensili), la disponibilità, lo spedito, le scadenze, gli inevasi ecettera. Si include anche la possibilità di notifica quando un nuovo report si aggiunge o viene modificato.

Il problema che sorge è quindi la mancanza di un sistema portatile e di facile installazione che interroghi il database gestionale, il quale ,in tempo reale fornisca una paginazione aggiornato dei dati d'interesse. Il tutto dovrà essere integrato facilmente nel progetto Budget che si sta sviluppando in parallelo. E' stato inoltre stabilito che il sistema dovesse anche tener conto della gestione di profilazione per utente. Lo scopo finale è quindi di avere un sistema robusto e di facile utilizzo che diminuisca i costi di mantenimento del codice. Questo perchè da un'analisi fatta dall'azienda stessa negli ultimi anni i costi di mantenimento dell'applicativo sono stati significanti. Tutto ciò è dovuto dal fatto che l'installazione e il funzionamento dell'applicativo preesistente doveva essere garantito su una larga gamma di sistemi operativi e browser.

A questo si aggiunge l'opportunità di includere un sistema di notificazione al verificarsi di un certo evento prestabilito, ed anche l'opportunita di un miglioramento dell'applicativo preesistente per quanto riguarda la gestione dell'inserimento e rimozione degli utenti con una profilazione dati per tipo utente.

2.2 Obiettivi

Gli obiettivi da raggiungere per la durata del progetto dello stage sono classificati in obbligatori e desiderabili con una struttura come segue.

La suddivisione degli obiettivi avviene per tipologia a secondo dell'importanza e vengono numerati in ordine crescente. Vengono rappresentati inoltre da una sigla formata da **OB-**[nuomero]-[tipologia]. Il numero è un valore progressivo che identifica l'obiettivo univocamente e la tipologia può essere **O** oppure **D** a seconda dell'importanza dell'obiettivo.

Gli obiettivi possono essere:

Obbligatori: rappresenta un requisito il cui soddisfacimento è fondamentale per raggiungere lo scopo prefissato nel progetto di stage;

Desiderabili: indica un requisito non necessario per raggiungere gli scopi prefissati nel progetto di stage ma che completerebbero maggiormente il risultato finale.

ID	IMPORTANZA	DESCRIZIONE
OB-1-O	Obbligatorio	Strudio e disegno del database per la permanenza dei dati che riguardano la gestione degli utenti, delle licenze e la gestione della tastiera Telegram.
OB-2-O	Obbligatorio	Acquisizione dei dati dal database gestionale, elaborazione e scrittura dati ETL
OB-3-O	Obbligatorio	Associazione utente B2B con l'utente telegram
OB-4-O	Obbligatorio	Gestione di profilazione delle reporistiche in base all'utente telegram.
OB-5-O	Obbligatorio	Creazione tabelle per le reporistiche in vari formati richiesti.
OB-6-O	Obbligatorio	Gestione attivazione utente generico da un amministratore. Gestione di inserimento e rimozione degli utenti generici da parte dell'utente amministratore.
OB-7-O	Obbligatorio	Gestione delle licenze, rimozione e prolungamenti delle stesse.
OB-8-O	Obbligatorio	Gestione lista utenti in attesa .
OB-9-D	Desiderabile	Test di controllo per le formule riassuntive dell'applicativo preesistente.
OB-10-D	Desiderabile	Integrazione della documentazione con quella dell'applicativo preesistente e sucessive aggiunte di diagrammi dei casi d'uso.

2.3 Vincoli

Per l'esecuzione dello stage presso Sanmarco Informatica S.p.A sono stati fissati alcuni vincoli:

Per facilitare l'integrazione nel progetto Budget, l'applicativo è stato richiesto di essere sviluppato in Java mantenendo così una certa compatibilità con la piattaforma di base del progetto Budget. Un altro vincolo è quello di usare il sistema ETL (Extract, Transform, Load) per estrarre e trasformare i dati dal database gestionale in un altro database che comunicherà direttamente con il nostro applicativo. Questo vincolo è dovuto dal fatto che non è consentito effettuare delle modifiche direttamente al database gestionale dell'azienda. Un altro criterio da rispettare è stato l'uso dell'IDE Eclipse Neon per la scrittura e la compilazione del codice sorgente. E' stato inoltre richiesto di usare RTC (Rational Team Concert), integrato nell'IDE eclipse per gestire la parte di versionamento del codice. Il tutor aziendale sempre attraverso l'RTC si impegnava ad assegnare gli sprint in corso e da concludere durante il progetto. Per la parte di pianificazione e gestione delle attività è stato richiesto di usare Asana.

2.4 Pianificazione

Per la pianificazione dello stage è stato suddiviso il lavoro in diverse attività assegnando una durata temporale idonea, tenendo conto dei limiti di tempo previsti di massimo 308 ore. Lo stage è stato suddiviso nelle seguenti attività:

* **Studio introduttivo delle tecnologi principali coinvolte :** in questo periodo dello stage è stato previsto lo studio e la configurazione di Eclipse con tutti i suoi plug-in necessari per avviare il progetto. Lo studio del tool DBeaver che è stato utilizzato per l'organizzazione e la gestione della base di dati e lo studio dell'architettura di base del Bot telegram dove si baserà la parte fron-end dell'applicativo.
* **Studio di fattibilità per la realizzazione del sistema :** in questo periodo è stato analizzato uno studio di fattibilità per capire se sia effettivamente possibile fornire un sistema sempre a portata di mano, robusto e dinamico, capace di fornire le funzionalità presenti nell'applicativo preesistente.
* **Analisi dei requisiti :** questa periodo rappresenta uno studio delle funzionalità che il software dovrebbe soddisfare per essere considerato funzionale dall'azienda.
* **Progettazione dell'architettura BI :** stabilire come saranno effettuate le query sul Data Base Postgres e come saranno presentati i dati elaborati sull'interfaccia front-end. Questa attività si suddivide in due parti, la prima parte riguarda la strutturazione delle query, ovvero dare una struttura ben precisa di

come i dati saranno interrogati dal database gestionale. Questo processo avviene attraverso le trasformazioni ed estrazioni continui ETL (Extract, Transform, Load) dal database gestionale dell'azienda al nostro database Postgres poiché non è consentito interrogare direttamente il database gestionale.
La seconda parte invece, riguarda l'attività di elaborazione dei dati ottenuti precedentemente dalla base dati e costruendo (nella maggior parte dei casi) una tabella con colonne che rappresentano i campi di interesse. Tipicamente le tabelle vengono salvate sotto forma di documenti PDF e spedite all'interfaccia front-end dove l'utente può consultarli, controllandoli periodicamente se sono in linea con le aspettative.

* **implementazione del sistema :** in questo periodo si prevede l'acquisizione dati con successiva scrittura attraverso ETL e l'implementazione dei servizi back-end che si occuperanno di riportare in modo efficiente ed efficace i dati richiesti dall'utente. Un aspetto importante in questo processo è quello di tener conto dei tempi di risposta al lato client.

* **Test e sperimentazione del sistema :** In questo periodo si prosegue con i test di correttezza e validazione del sistema e delle modifiche apportate per assicurare che rispettino le aspettative previste.

* **Documentazione :** Questo processo comprende l'intera attività di documentazione effettuata durante il periodo di stage. Al contrario delle altre, questa è un'attività trasversale che riguarda in parte ogni altra attività e continua per l'intera durata dello stage.

2.5 Ambiente di lavoro

All'inizio dell'attività di stage, dall'azienda Sanmarco Informatica S.p.A. ho avuto accesso a diverse risorse dell'azienda per permettermi di eseguire il lavoro al meglio.
Le risorse condivise dall'azienda si suddividono in risorse hardware e risorse software, con i programmi messi a disposizione dall'azienda, e risorse informative, che comprendono i materiali di studio forniti inizialmente dall'azienda.

2.5.1 Risorse hardware

Il team NextBI che fa parte nell'ambiente di Ricerca e Sviluppo dell'azienda Sanmarco Informatica S.p.A. mi ha seguito durante lo stage rendendomi il lavoro più facile con la loro collaborazione e disponibilità di formazione.

Mi è stato consegnato un PC aziendale già configurato con gli ambienti di lavoro aggiornati e alcuni software preinstallati per facilitare la fase di configurazione dell'ambiente.
Oltre al PC aziendale, mi è stato consentito l'utilizzo di una macchina virtuale per effettuare dei test necessari, molto utile soprattutto per l'esecuzione di ETL che richiedono tempi di esecuzione molto lunghi.

2.5.2 Risorse software

Ho avuto accesso al repository aziendale basato su RTC per gestire il versionamento del codice. I'RTC è stato configurato con l'IDE di Eclipse per garantire la compatibilità

del sistema di versionamento già present nell'applicativo preesistente.
Il PC aziendale inoltre, aveva preinstallato la maggior parte dei software utili per effettuare il lavoro con versioni aggiornati.

Inoltre mi è stato consentito installare anche nuovi software, tools o librerie se necessari, utili in particolati circostanze.

2.5.3 Risorse informative

Mi sono state fornite diverse risorse informative tra cui diverse note, manuali di programmazione e la relativa documentazione dell'applicativo preesistente nel quale si va ad integrare il progetto di stage. Quest'ultimo ha favorito l'apprendimento corretto del funzionamento dell'applicativo preesistente a fine di ammortizzare i tempi di progettazione.

2.6 Tecnologie usate

Sono varie le tecnologie usate durante lo stage. In questo paragrafo vengono descritte le diverse tecnologie usate con una breve presentazione.

1. **Java:** Java è un linguaggio di programmazione ad alto livello, orientato agli oggetti e a tipizzazione statica, specificatamente progettato per essere il più possibile indipendente dalla piattaforma di esecuzione.
   La sua semplicità unita all'essere un linguaggio multi piattaforma lo rende molto usato e fa si che disponga di un'elevata quantità di librerie facilmente integrabili per le attività varie.
   Uno dei principi fondamentali del linguaggio Java è espresso dal motto WORA (write once, run anywhere, ossia "scrivi una volta, esegui ovunque"): il codice compilato che viene eseguito su una piattaforma non deve essere ricompilato per essere eseguito su una piattaforma diversa. Il prodotto della compilazione è infatti in un formato chiamato bytecode che può essere eseguito da una qualunque implementazione di un processore virtuale detto Java Virtual Machine.
2. **XML (Extensible Markup Language) :** XML è un linguaggio di markup che definisce regole per la codifica di documenti in un formato comprensibile sia se letto da un umano che da una macchina. Lo scopo principale del formato è concentrato sulla semplicità, generalità e usabilitaà generale. Il formato permette quindi di definire tag personalizzati per i vari campi e mantenere un output che sia analizzabile in maniera automatica e manuale. Il formato è stato usato perchè già integrato all'interno del Plug-in MyBatis per definire le tabelle del database.
3. **PostgresSQL :** PostgreSQL è un completo DBMS (modello di base di dati) ad oggetti rilasciato con licenza libera (stile Licenza BSD).
   PostgreSQL è una reale alternativa sia rispetto ad altri prodotti liberi come MySQL, Firebird SQL che a quelli a codice chiuso come Oracle, IBM o DB2 ed offre caratteristiche uniche nel suo genere che lo pongono per alcuni aspetti all'avanguardia nel settore dei database.
   PostgresSQL è stato utile nel progetto poichè ha permesso di collegare diversi database e farli comunicare tra loro con un'interfaccia facile da manovrare.

PostgreSQL permette anche di definire nuovi tipi basati sui normali tipi di dato SQL, permettendo al database stesso di comprendere dati complessi.
PostgreSQL, inoltre, permette l'ereditarietà dei tipi, uno dei principali concetti della programmazione orientata agli oggetti
In PostgreSQL si può implementare la logica in uno dei molti linguaggi supportati.

4. **Notepad++ :** Notepad++ è un text editor open source mirato alla modifica di codice sorgente. Il programma non dispone di tutte le capacità di un vero IDE ma presenta funzionalità più basilari come syntax highlighting (evidenziazione della sintassi) per vari linguaggi tra cui anche Java, ricerca avanzata anche tramite espressioni regolari e la possibilità di aggiungere diversi plugin per espanderne le capacità. Notepad++ si è rivelato molto utile sia come semplice text editor ma anche per permettere rapide modifiche al codice grazie alla maggiore rapidità presentata rispetto all'Eclisse.

5. **Astah Community Astah Community:** Astah è un programma per la modellazione di schemi UML (Unified Modeling Language) cioè rispettanti degli standard industriali per i modelli general-purpose per l'ingegneria del software. Il programma è stato scelto perchè ritenuto abbastanza semplice da usare per lo scopo necessario dopo esperienze precedenti ed è stato usato per creare alcuni diagrammi durante la fase di progettazione.

6. **Evernote :** Evernote è un programma multi piattaforma mirato alla creazione di note, la loro organizzazione, archiviazione e condivisione. Dispone sia di client installabile sia di un'interfaccia web e permette anche di condividere piccoli file. L'uso del programma è stato richiesto da parte del tutor ed è servito per parte del processo di documentazione. Su di esso sono stati infatti tenute cronologie, appunti e tracce delle decisioni che sono state fatte man mano durante il proseguimento dello stage.

7. **Visual Studio Code :** Visual Studio Code è un editor di codice sorgente sviluppato da Microsoft compatibile cin diversi sistemi operativi. Questo editor supporta vari linguaggi di programmazione, tra cui anche JavaScript e Java. Visual Studio Code permette installare varie estensioni, con la possibilità di fare ciò direttamente dal programma.
Visual Studio è stato usato principalmente per facilitare lo sviluppo front-end grazie alle sue numerose estensioni disponibili facilmente instancabili.

8. **Telegram :** Telegram è un servizio di messaggistica istantanea basato su cloud ed erogato senza fini di lucro dalla società Telegram LLC. I client ufficiali di Telegram sono distribuiti come software libero per diverse piattaforme.

   Da giugno 2015 Telegram ha introdotto una piattaforma per permettere, a sviluppatori terzi, di creare i Bot. I Bot sono degli account Telegram, gestiti da un programma, che offrono molteplici funzionalità con risposte immediate e completamente automatizzate. Grazie a questa funzionalità, il Bot telegrma è diventata la piattaforma dove si basa la maggior parte dell'applicativo sviluppato durante il periodo dello stage.

9. **FileZilla :** FileZilla Client è un software libero multipiattaforma che permette il trasferimento di file in Rete attraverso il protocollo FTP. Il programma è disponibile per GNU/Linux, Microsoft Windows, e macOS. Tra i vari protocolli

supportati, oltre all'FTP vi è l'SFTP, e l'FTP su SSL/TLS. Il codice sorgente di FileZilla è disponibile sul sito SourceForge e, in alcuni casi, anche al momento dell'installazione del programma. Il progetto fu nominato Progetto del Mese nel novembre del 2003.
Questo software è stato usato nel periodo dello stage per poter permettere dei trasferimenti di diversi file dal server dell'azienda Sanmarco Informatica a quello dei clienti dove è stato installato la demo.

Altre tecnologie ....

Capitolo 3

Definizione del problema

Breve introduzione al capitolo

3.1 Sempre più portabilità

Quando le attività da gestire in un'azienda diventano tante, si ha la necessità di tenerle sotto controllo molte volte durante la giornata e questo purtroppo richiede quasi sempre un PC o tablet per poterci accedere attraverso una pagina web. Questo problema lo ritroviamo in tanti altri sistemi dove si richiede l'utilizzo di un'interfaccia web per interagire. Eliminando l'interfaccia web sostituendola con un applicazione installabile a volte non è la soluzione migliore perché questo significa dover fornire una compatibilità a tutte le piattaforme e sistemi operativi disponibili altrimenti si limiterebbe il supporto delle funzionalità solo su una specifica piattaforma o sistema operativo. Bisognerebbe quindi trovare un modo per rendere l'applicativo un'applicazione portabile, cioè senza il bisogno dell'installazione e questo problema viene affrontato nel prossimo capito.

3.2 Il problema della notificazione

Nell'applicativo preesistente non è presente la possibilità di notificare l'utente in tempo reale quando succede qualcosa su un certo evento programmato. Questa estensione sarebbe molto apprezzata nel nuovo applicativo. Fornire un sistema di notificazione significa creare degli eventi e restare in ascolto su di essi finché non succeda qualcosa per scaturire l'evento.

Negli ultimi tempi di internet, queste funzionalità hanno preso piede in diversi ambiti. Oggigiorno grazie alle "Notification API" siamo in grado di inviare delle notifiche sfruttando il sistema operativo che sta usando il nostro utente. Possiamo creare ad esempio una chat online, come Slack, e sfruttare le notifiche per avvertire l'utente che ha ricevuto un nuovo task da portare al termine, oppure possiamo creare all'interno della nostra applicazione web un ToDo List, come "Asana", per ricordare all'utente che si sta avvicinando la scadenza di un determinato task.
Questo tipo di notifiche sono molto interessanti per diverse tipologie di applicazioni web ma il primo problema da affrontare con le applicazioni web è la compatibilità delle

"Notifications API" con i vecchi browser essendo le Notifications API una tecnologia relativamente recente. Anche questo problema sarà affrontato nel prossimo capitolo.

3.3 L'aggiornamento in real time

Le aziende che si occupano di gestionale si trovano quasi sempre a gestire dati molto importanti e sensibili di altre aziende. Per questo motivo il software gestionale limita l'accesso e modifica dei dati nel database secondo la propria politica di gestione. Per poter interfacciarsi con questo problema bisogna trovare un sistema che faccia una specie di "backup" delle tabelle di interesse del gestionale sul quale effettuarci le query e successivamente riportarle nel database gestionale di origine. A questo punto si presenta un problema fondamentale: cioè capire ogni quanto tempo bisogna fare l'aggiornamento delle tabelle dal database gestione. Questo problema influisce direttamente sulle performance dell'applicativo poichè, come specificato sopra, i dati devono essere aggiornati il più possibile nel momento in cui si effettua una richiesta dall'interfaccia utente. Ma dall'altra parte, aggiornando le tabelle ogni secondo richiederebbe tanta velocità di calcolo e rischio di deformazione dei dati. Bisogna dunque trovare un compromesso ragionevole.

3.4 Trasformazione dei dati

Un altro problema da affrontare è quello di effettuare delle trasformazioni dei dati in altri formati. Spesso nei sistemi gestionali è richiesta una rappresentazione dei dati dal database in diversi formati come ad esempio in XML, Doc, PDF eccetera.
Come descritto nel prossimo capito questo problema vine affrontato con il sistema ETL, più specificamente verrà usato il software di PDI (Pentaho Data Integration) chiamto Kettle.

3.5 Analisi del problema e soluzione

3.5.1 Studio iniziale del problema di portabilità

Dopo una attenta analisi per quanto riguarda il problema della portabilità dell'applicativo, è stato deciso sin dal primo tentativo di affiancare l'interfacci web esistente con una mobile. Questa decisione si basa sul fatto che avere un'interfaccia mobile è sempre comodo da usare e portare con se, dato che al giorno d'oggi, tutti possediamo uno smartphone personale sempre con noi. E' stata esclusa subito invece la possibilità di creare un'applicazione mobile da zero. Questo perchè come descritto nel primo capitolo si cerca di avere una soluzione capace di fornire un'applicativo dinamico, di facile utilizzo, che includa un sistema di notificazione per gli eventi preimpostati, che non abbia bisogno di essere installato e sopratutto che non abbia nessun tipo di dipendenza dal sistema operativo o browser.
In secondo luogo è stata analizzata la possibilità di utilizzare una piattaforma già pronta open source che implementi un modulo di messaggistica chat, capace di ospitare il nostro sistema di notificazione attraverso l'invio di messaggi, evitando cosi di realizzare uno tutto da zero.

Navigando su internet si trovano decine di chat application open source che possono andar bene al caso nosto. Tra le più utilizzate sono:

* Slack
* RocketChat
* IRC
* Let's Chat
* Telegram

Scelta della Chat Application

Valutando le funzionalità offerte dalla lista delle Chat Application è stato scelto Telegram come servizio di messaggistica di appoggio. Recentemente Telegram ha introdotto una nuova piattaforma per permettere agli sviluppatori di creare i Bot. I Bot sono degli account Telegram, gestiti da un programma, che offrono molteplici funzionalità con risposte immediate e completamente automatizzate.

Uno dei motivi principali per cui è stata presa questa decisione è la guida completa e la dettagliata documentazione del codice sorgente fornita dai membri Telegram. Grazie ai numerosi sviluppatori e membri attivi telegram, è sempre più facile trovare una risposta ai problemi nei appositi forum oppure contattando direttamente i loro membri.
Telegram supporta lo sviluppo con la maggior parte dei linguaggi di programmazione tradizionali (non solo OOP), tra i quali Java. Questo ha reso ancora più facile la nostra scelta nel progetto.

qui verrano aggiunti altri dettagli della scelta telegram .....

3.5.2 Il problema dell'aggiornamento dei dati in real time

Per ovviare questo problema è stato deciso di realizzare degli script i quali fanno partire dei processi per aggiornare le tabelle di interesse sistematicamente in un orario prefisato. L'applicazione Kettle fornito dall'impresa Pentaho, è stato un tool programmabile molto utile nel gestire i processi di lettura ed aggiornamento delle tabelle. In queto modo i dati saranno aggiornati nel database dell'applicativo e poichè le operazione effettuate dall'utente sono prevalentemente di lettura (dall'interfaccia web invece non è cosi), il problema della mutua esclusione non si presenta in questo scenario.

3.5.3 L'associazione B2B - Telegram

Come vedremo nella progettazione le operazioni che andranno a scrivere sul database sono quelle che si occupano di creare un'associazione tra l'utente B2B e quello dell'account Telegram. Il database sul quale verrano memorizzati gli account degli utenti è diverso da quello che conterrà i dati del B2B. Questa separazione servirà per garantire un'identificazione univoca degli utenti B2B con quelli registrati con il Bot Telegram. Inoltre verrà utilizzato lo stesso database che si occupa della gestione degli utenti, per gestire anche le licenze e la loro validità per ciascun utente. In questo caso l'unico utente che modificherà e attiverà le licenze sarà l'utente amministratore per cui siamo

sicuri che non si presenterà nessun problema di sincronizzazione e modifiche multiple da parte di più utenti.

3.5.4 La soluzione scelta per la trasformazione dei dati

Per quanto riguarda le trasformazioni ETL dei dati è stato scelto di utilizzare il software Kettle il quale è un software molto affidabile e presenta un'interfaccia molto intuitiva. Grazie a Kettle è stato possibile estrarre i dati dalla sorgentie gestionale ed elaborarli ulteriormente facendoli subire le trasformazioni desiderate. Le più comuni trasformazioni sono state: la normalizzazione dei dati, l'eliminazione dei duplicati, la derivazione dei nuovi calcolati, il raggruppamento degli stessi selezionando quelli di interesse eccetera.

Capitolo 4

Analisi dei requisiti

Breve introduzione al capitolo

4.1 Casi d'uso

In questa sezione vengono trattati i casi d'uso riguardanti l'interazione dell'utente con l'interfaccia visiva dell'applicativo. Con i casi d'uso si vuole mostrare una panoramica delle funzionalità dell'applicativo. Vengono inseriti gli Use Cases DiagramG e la loro descrizione secondo le specifiche UML2.0G. L'attore principale è sempre l'utente generico autenticato oppure l'utente amministrativo autenticato.

Pagina iniziale

Attori primari: Utente autenticato

Descrizione: L'utente generico può visualizzare la lista degli ordini, può visualizzare le scadenze, lo spedito, la disponibilità degli ordini, gli inevasi e lo scadenzario. L'utente amministratore (che è una generalizzazione dell'utente generico) può gestire gli utenti, attivando una licenza, rimuovere gli utenti, associare un utente con un'account B2B e rimuovere un'associazione.

Precondizione: Il sistema funziona correttamente e visualizza la pagina principale correttamente.

Postcondizione: Il sistema ha ricevuto uno dei comandi eseguiti dall'utente ed è stato scaricato il file che visualizza le informazioni richieste dall'utente oppure è stato aperta la pagina dell'amministrazione degli utenti nel caso dell'utente amministratore.

Flusso base degli eventi:

* L'attore visualizza la lista del tipo di ordine da selezionare (UC1);
* L'attore può visualizzare la lista del tipo di scadenze da selezionare (UC2);
* L'attore può visualizzare la lista del tipo di spedito da selezionare (UC3);

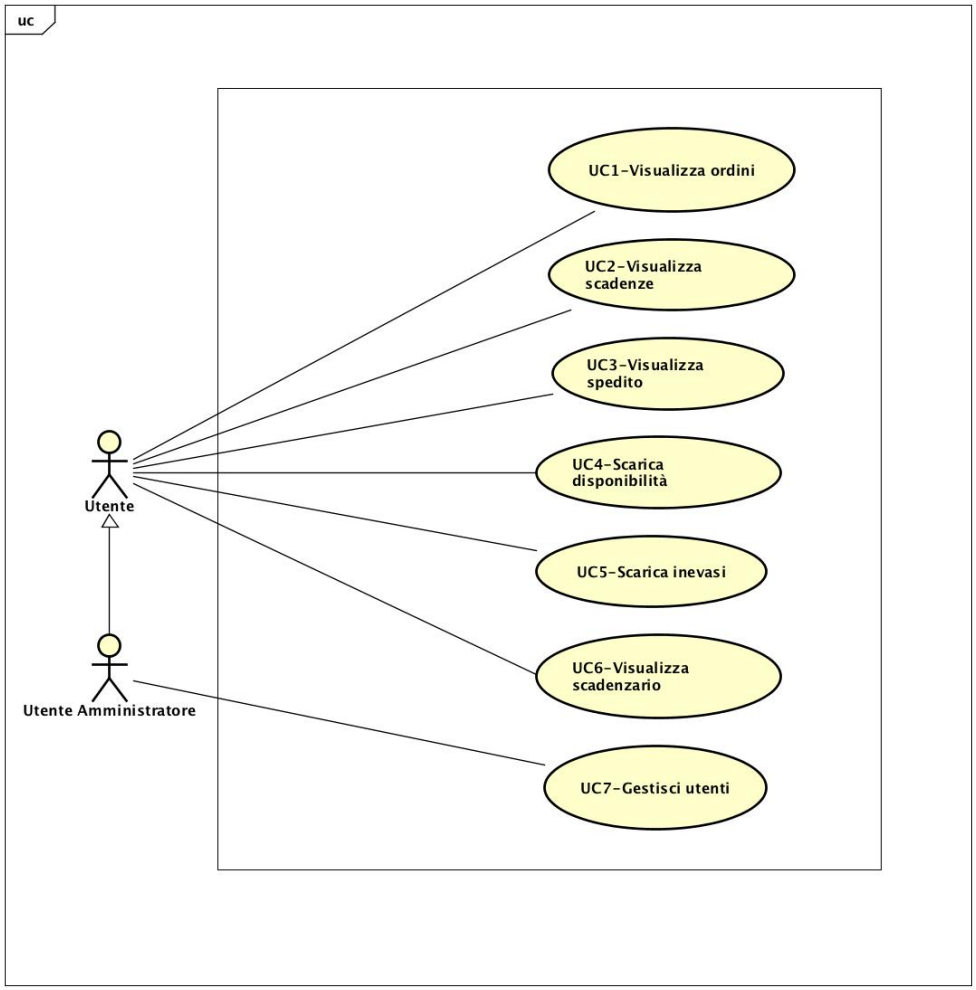


Figura 4.1: Use Case - UC0: Scenario principale

* L'attore può scaricare il documento aggiornato del report di Disponibilità (UC4);
* L'attore può scaricare il documento aggiornato del report Inevaso (UC5);
* L'attore può visualizzare la lista del tipo di scadenzario da selezionare (UC6);
* L'attore (utente amministratore) può visualizzare la lista dei comandi per gestire l'associazione tra utenti, l'eliminazione e la gestione delle licenze (UC7);

Amministrazioni utenti

Attori primari: Utente amministratore autenticato

Descrizione: L'utente amministratore può visualizzare la lista degli utenti che risultano attivi con una licenza, visualizzare gli utenti che sono in attesa ad essere registrati nel sistema (cioè gli utenti che hanno fatto richiesta per essere registrati

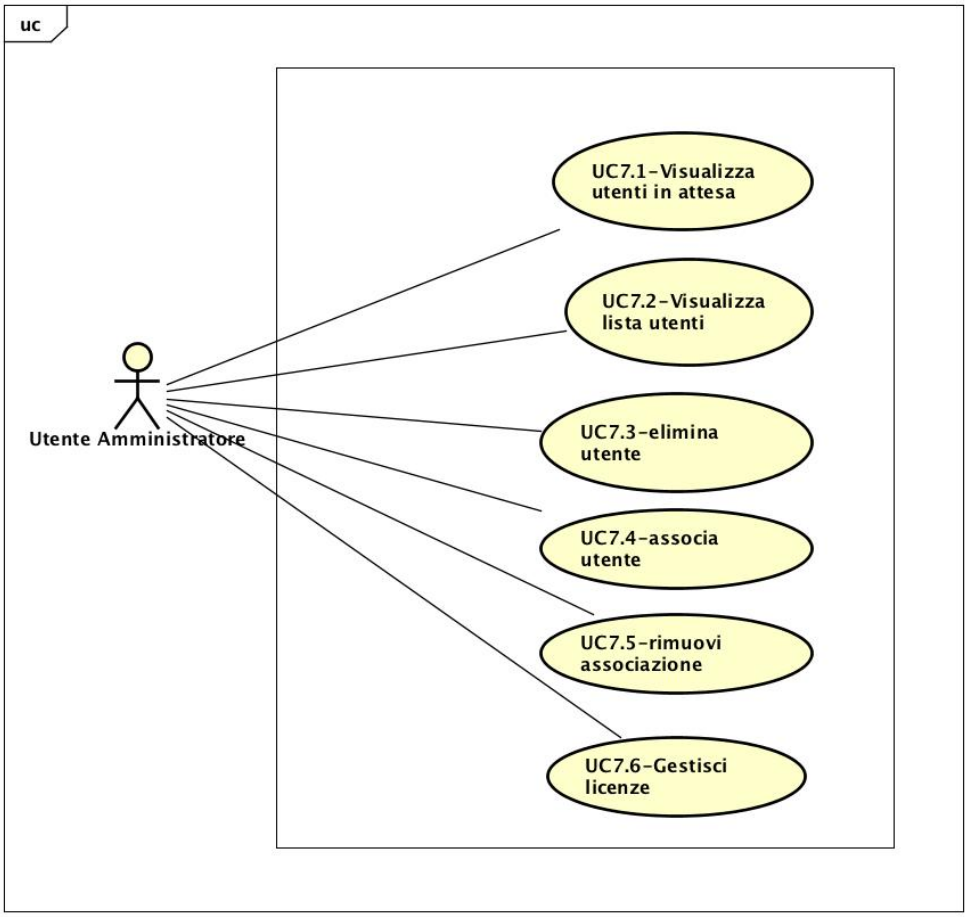


Figura 4.2: Use Case - UC0: Scenario principale

con una licenza ma che non sono ancora accettati da un amministratore). L'utente amministratore può eliminare un utente generico può gestire la licenza, inserendo una nuova oppure visualizzare lo stato di quelle attive. Infine l'utente amministratore può effettuare un'associazione tra l'utente B2B ed il suo account telegram oppure può eliminare un'associazione precedentemente creata.

Precondizione: Il sistema funziona correttamente e visualizza la pagina all'amministrazione utenti correttamente.

Postcondizione: L'utente amministratore è stato in grado ad effettuare correttamente un'operazione da lui voluta nella pagina di gestione utenti.

Flusso base degli eventi:

* L'utente amministratore visualizza la lista degli utenti in attesa ad essere registrati (UC7.1);
* L'utente amministrativo visualizza la lista degli utenti registrati correttamente

con una licenza valida (UC7.2);

* L'utente amministrativo elimina un utente rimuovendo anche la relativa licenza (UC7.3);
* L'utente amministrativo associa un utente B2B con il suo account telegram (UC7.4);
* L'utente amministrativo rimuove l'associazione di un utente B2B con l'account telegram (UC7.5);
* L'utente visualizza la lista dei comandi per l'amministrazione di un utente generico (UC7.6);

Ordini del giorni

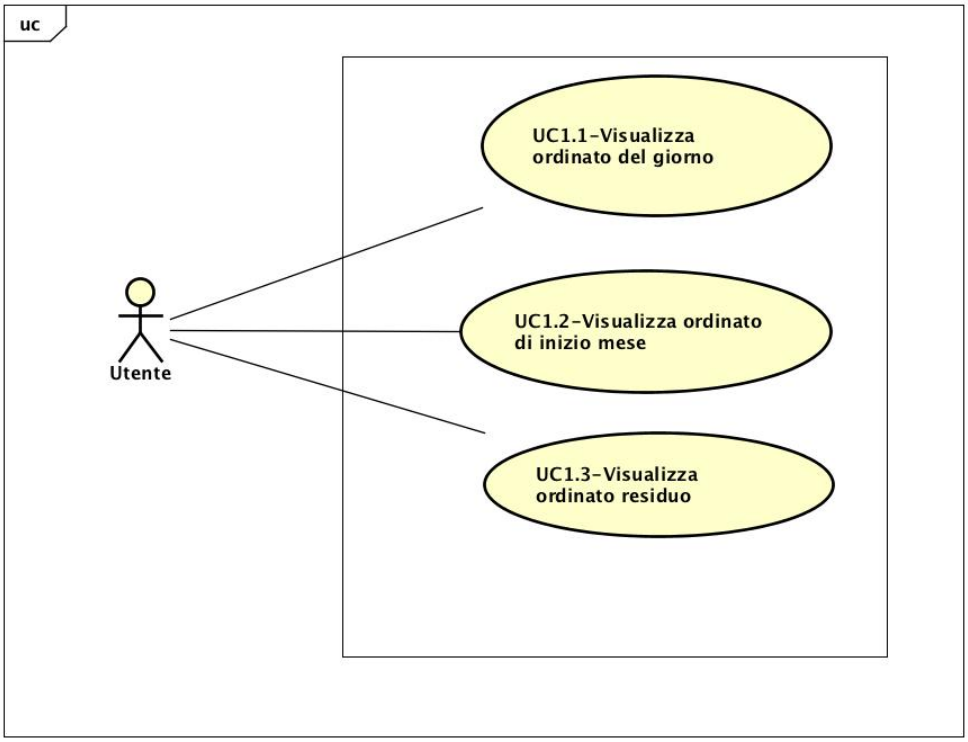


Figura 4.3: Use Case - UC0: Scenario principale

Attori primari: Utente generico ed utente amministratore autenticati

Descrizione: L'utente (generico oppure amministratore) può scegliere di visualizzare il report "ordinato del giorno", "ordinato di inizio mese" o "ordinato residuo".

Precondizione: Il sistema funziona correttamente e visualizza la pagina per poter scaricare gli ordinati correttamente.

Postcondizione: L'utente generico oppure amministratore è stato in grado di visualizzare correttamente un "ordinato" da lui voluto nella pagina di visualizzazione ordini.

Flusso base degli eventi:

* L'utente visualizza i valori inerente all'ordinato del giorno (UC1.1);
* L'utente visualizza i valori inerente all'ordinato di inizio mese (UC1.2);
* L'utente visualizza i valori inerente all'ordinato residuo (UC1.3);

Scadenze

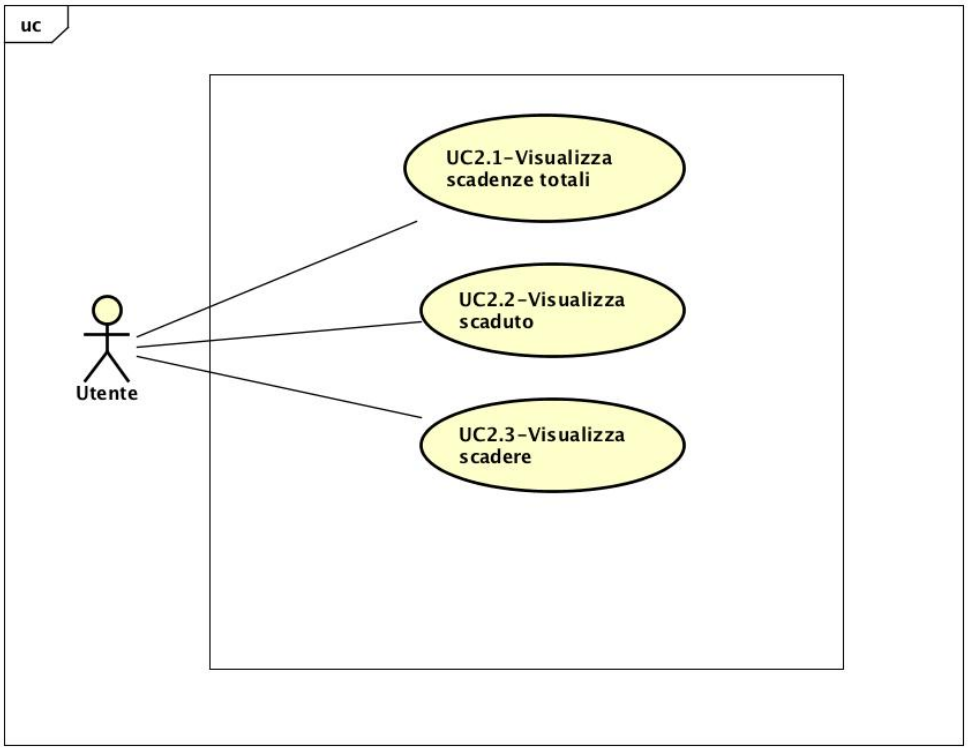


Figura 4.4: Use Case - UC0: Scenario principale

Attori primari: Utente generico ed utente amministratore autenticati

Descrizione: L'utente (generico oppure amministratore) può scegliere di visualizzare l'ammontare delle "scadenze totali", dello "scaduto" oppure "scadere".

Precondizione: Il sistema funziona correttamente e visualizza la pagina per poter visualizzare le scadenze correttamente.

Postcondizione: L'utente generico oppure amministratore è stato in grado di visualizzare correttamente le scadenze da lui volute nella pagina di visualizzazione scadenze.

Flusso base degli eventi:

* L'utente visualizza i valori inerenti alle scadenze totali (UC2.1);
* L'utente visualizza i valori inerenti allo scaduto (UC2.2);
* L'utente visualizza i valori inerenti allo scadere (UC2.3);

Gli spediti

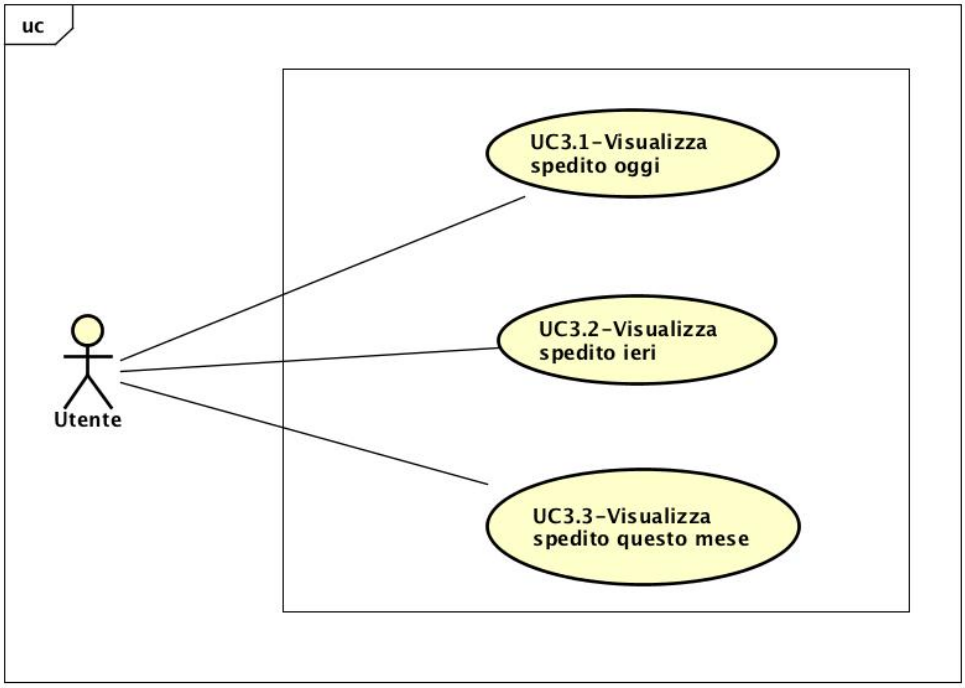


Figura 4.5: Use Case - UC0: Scenario principale

Attori primari: Utente generico ed utente amministratore autenticati

Descrizione: L'utente (generico oppure amministratore) può scegliere di visualizzare l'ammontare dello "spedito oggi", dello "spedito ieri" oppure "spedito questo mese".

Precondizione: Il sistema funziona correttamente e visualizza la pagina per poter visualizzare gli spediti correttamente.

Postcondizione: L'utente generico oppure amministratore è stato in grado di visualizzare correttamente gli spediti da lui volute nella pagina di visualizzazione scadenze.

Flusso base degli eventi:

* L'utente visualizza i valori inerenti allo spedito oggi (UC3.1);
* L'utente visualizza i valori inerenti allo spedito ieri (UC3.2);
* L'utente visualizza i valori inerenti allo spedito questo mese (UC3.3);

Scadenzario

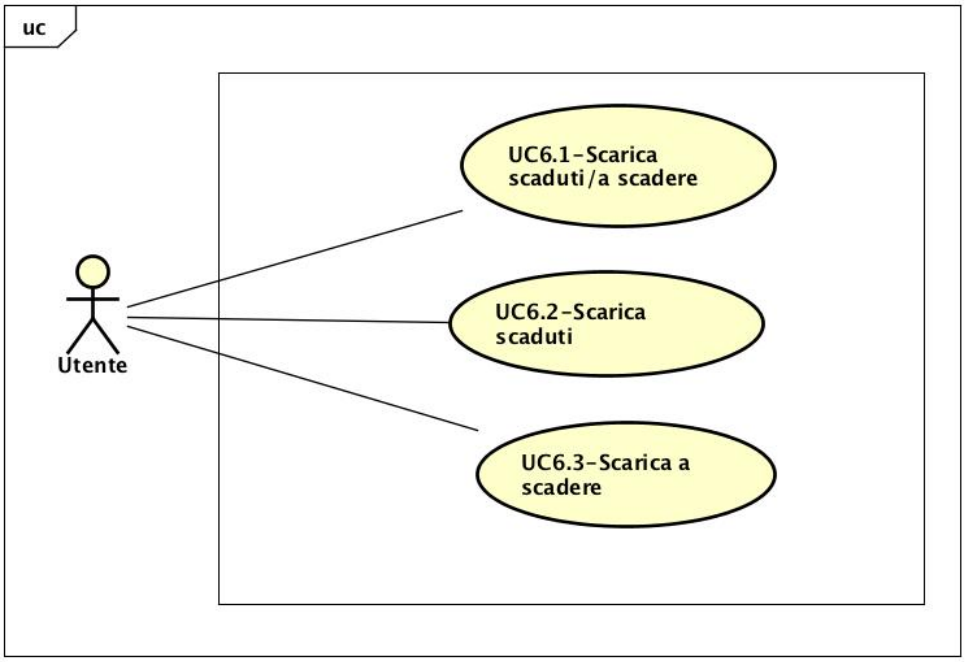


Figura 4.6: Use Case - UC0: Scenario principale

Attori primari: Utente generico ed utente amministratore autenticati

Descrizione: L'utente (generico oppure amministratore) può scegliere di scaricare il documento che rappresenta il report di "scaduti/a scadere", "scaduti" oppure "a scadere".

Precondizione: Il sistema funziona correttamente e visualizza la pagina per poter scaricare lo scadenzario correttamente.

Postcondizione: L'utente generico oppure amministratore è stato in grado di scaricare correttamente lo scadenzario da lui volute nella pagina di visualizzazione scadenzario.

Flusso base degli eventi:

* L'utente scarica il report inerente agli scaduti/a scadere (UC6.1);

* L'utente scarica il report inerente agli scaduti (UC6.2);
* L'utente scarica il report inerente allo scadere (UC6.3);

Requisito	Descrizione	Use Case
RF-0	L'attore può visualizzare la pagina principale per poter scegliere uno delle funzionalità presenti	UC0
RF-1	L'attore visualizza la lista del tipo di ordine da selezionare (l'ammontare degli ordini in euro)	UC1
RF-2	L'attore può visualizzare l'ordinato del giorno	UC1.1
RF-3	L'attore può visualizzare l'ordinato di inizio mese	UC1.2
RF-4	L'attore può visualizzare l'ordinato residuo	UC1.3
RF-5	L'attore visualizza la lista del tipo di scadenze da selezionare (l'ammontare delle scadenze in euro)	UC2
RF-6	L'attore può visualizzare le scadenze totali	UC2.1
RF-7	L'attore può visualizzare lo scaduto	UC2.2
RF-8	L'attore può visualizzare lo scadere	UC2.3
RF-9	L'attore visualizza la lista del tipo di spedito da selezionare (l'ammontare dello spedito in euro)	UC3
RF-10	L'attore può visualizzare lo spedito di oggi	UC3.1
RF-11	L'attore può visualizzare lo spedito di ieri	UC3.2
RF-12	L'attore può visualizzare lo spedito del mese	UC3.3
RF-13	L'attore può scaricare il documento aggiornato del report di Disponibilità	UC4
RF-14	L'attore può scaricare il documento aggiornato del report Inevaso	UC6
RF-15	L'attore può visualizzare la lista dei comandi per gestire l'associazione tra utenti, l'eliminazione e la gestione delle licenze	UC7
RF-16	L'attore può visualizzare gli utenti in attesa di registrazione	UC7.1
RF-17	L'attore può visualizzare la lista gli utenti già registrati	UC7.2
RF-18	L'attore può eliminare un utente	UC7.3
RF-19	L'attore può associare un utente B2B con il suo account telegram	UC7.4
RF-20	L'attore può rimuovere l'associazione di un utente B2B con il suo account telegram	UC7.5
RF-21	L'attore può gestire la licenza di un utente	UC7.6
RF-22	L'attore può visualizzare le licenze degli utenti	UC7.7
RF-23	L'attore può inserire una licenza	UC7.8

Tabella con requisiti qualitativi

Requisito	Descrizione	Use Case
RQ-1	Il codice sviluppato deve essere versionato con il sistema RTC integrato nello strumento Eclipse	Azienda
RQ-2	Si deve usare il software Kettle per la gestione ETL	Azienda

Tabella con i vincoli

Requisito	**Descrizione**	**Use Case**
RV-1	Il prodotto deve essere sviluppato secondo rispettando gli standard ISO aziendali	Azienda
RV-2	L'interfaccia front-end deve essere sviluppata, basandosi sulla piattaforma telegram per sfruttare le notifiche push	Azienda
RV-3	La logica di BI deve essere sviluppata come web-services sul server Java Tomcat	Azienda
RV-4	Come DBMS si deve usare PostgresSQL	Azienda

Capitolo 5

Progettazione e codifica

In questo capitolo vengono descritte le principali scelte progettuali, la struttura del database, la struttura del server e dell'applicativo client-side.

5.1 Database

Per gestire l'area di amministrazione degli utenti è stato costruito il database locale di amministrazione chiamato "data", appunto perché dovrà rappresenta tutti i dati amministrativi dell'utente, la gestione dell'associazione tra utenti b2b con quelli di telegram, le licenze degli utenti e tutta la gestione della tastiera telegram. La creazione e la gestione della tastiera di telegram è automatizzata dalla tabella Keyboard, ovvero per creare un nuovo tasto associato ad una funzionalità, bisogna semplicemente scrivere sulla tabella Keyboard il nome del tasto, su quale riga si vuole posizionare e il nome del metodo che verrà chiamato quando si preme quel bottone. Questo sistema di gestione di tastiera è molto comodo per tenere una separazione tra i vari componenti che implementano una logica diversa. In questo modo è molto facile estendere l'applicativo con altre funzionalità modificando semplicemente una sola riga del database.

5.1.1 Struttura

In questa sottosezione verranno elencate tutte le tabelle che rappresentano la gestione dell'area amministrativa, spiegando il loro uso e funzionamento in modo da comprendere l'utilizzo che ne è stato fatto nel progetto per il salvataggio dei dati.

User

In questa tabella vengono salvati i dati anagrafici dell'utente telegram,il suo ID identificativo univoco e il suo ruolo aziendale.

User ha i seguenti campi:

* **id:** id univoco per identificare un univoco utente telegram;
* **first_name:** il nome dell'utente telegram;

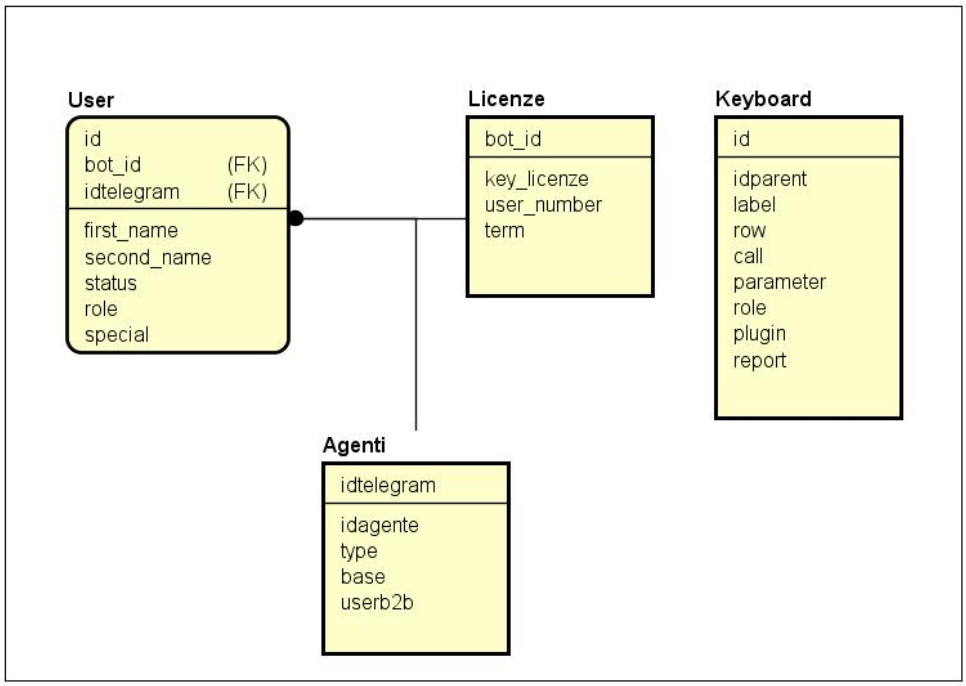


Figura 5.1: ER Database

* **second_name** il cognome dell'utente telegram;
* **status:** lo status rappresenta se l'utente è abilitato "allow" oppure no, "deny";
* **role:** role rappresenta il ruolo aziendale dell'utente;
* **special:** special è un campo speciale utilizzato solo lato sviluppo per dare il massimo dei permessi agli sviluppatore, utile in fase di testing;

Agenti

In questa tabella vengono salvati i dati riguardanti gli agenti, il loro Idtelegram, il loro tipo e la corrispondenza con il codice utente B2B.

La tabella Agenti presenta i seguenti campi:

* **id_telegram:** id univoco per identificare un univoco utente telegram;
* **id_agente:** id univoco per identificare un agente;
* **type:** rappresenta il tipo dell'agente;
* **base:** rappresenta la base di appartenenza dell'agente;
* **userb2b:** rappresenta il codice utente corrispondente in B2B;

Licenze
In questa tabella vengono salvati i dati riguardanti le licenze con le loro scadenze per ciascun utente.

La tabella Licenze presenta i seguenti campi:

* **bot_id:** botid rappresenta l'ID univoco per identificare un utente telegram;
* **key_licenze:** rappresenta la chiave di licenza per gli utenti telegram;
* **user_number** rappresenta il numero massimo di account utilizzabili;
* **term** rappresenta la data di scadenza della licenza;

Keyboard

In questa tabella vengono salvati i dati che riguardano la generazione della tastiera, la posizione e il nome del tasto.

La tabella Keyboard presenta i seguenti campi:

* **id:** id rappresenta l'ID associato al tasto univoco per identificare un utente telegram;
* **id_parent:** rappresenta la l'id del padre sul quale appendere il tasto;
* **label** il campo label rappresenta l'etichetta del bottone;
* **row** il campo row rappresenta il numero di riga dove si vuole aggiungere il tasto;
* **call:** rappresenta la chiamata al metodo da effettuare quando si preme il bottone;
* **parameter** parameter rappresenta i parametri del metodo da chiamare
* **role:** role rappresenta il ruolo dell'utente (admin oppure user);
* **plugin** rappresenta il nome del plugin associato se c'è ne sono;
* **report** rappresenta il nome del report;

5.2 Back-end

5.2.1 Server Telegram

Nel progetto è stato implementato il metodo di long polling per le notifiche di telegram. Questo significa che ci si collega periodicamente al server telegram per aggiornarsi delle nuove notifiche. Come si vede nella prima figura, l'utente invia un messaggio ad un altro utente (in questo caso chiamato Your Computer). Il messaggio arriva al server telegram, il quale manda la richiesta al destinatario, dopodichè avviene il polling del messaggio verso il server.
Per la risposta invece il messaggio viene mandato prima al server telegram, successivamente viene mandato dal server all'utente desiderato.
Non possiamo dire nulla invece al riguardo del server telegram (il quale lo consideriamo come un Black-Box), poichè il codice sorgente del server telegram non è disponibile ancora open source per il momento.

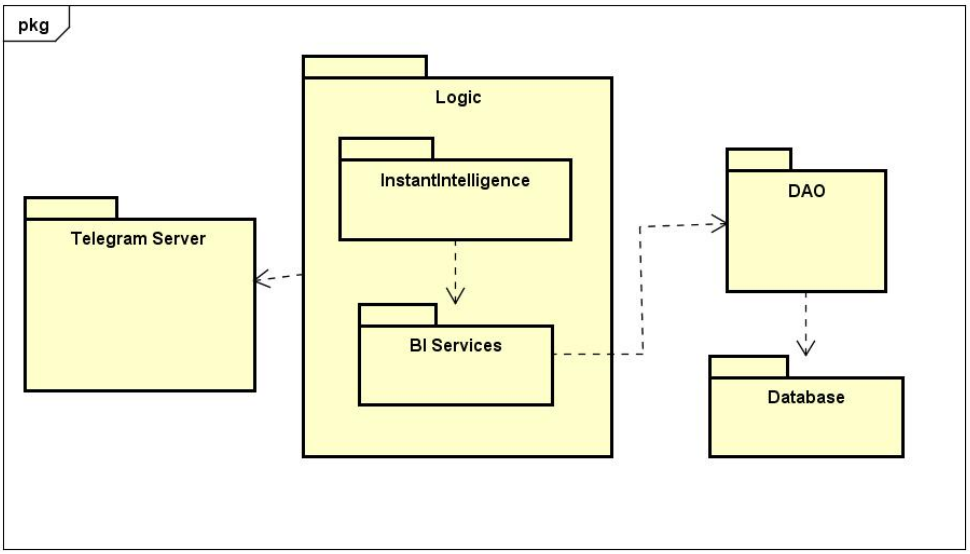


Figura 5.2: Diagramma Back-end

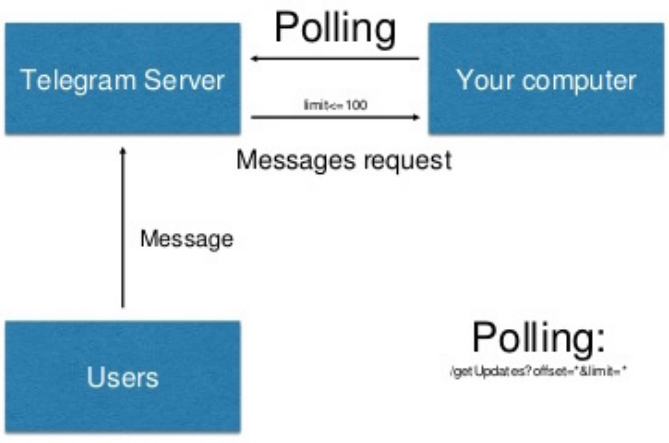


Figura 5.3: Diagramm Polling

5.2.2 InstantIntelligence

Il package InstantIntelligence invece è la parte dove vengono elaborate le richieste dell'utente in base al tasto premuto. In questo package viene effettuato il controllo dell'autenticazione dell'utente, ovvero viene controllato se l'utente abbia una licenza valida oppure è la prima volta che sta accedendo e quindi sta chiedendo di essere abilitato. Questo package si occupa anche della creazione e gestione della tastiera con i dati presi nel database Keyboard. Ogni richiesta fatta dall'utente, premendo un tasto oppure digitando un semplice comando passa per InstantIntelligence ed in base alla richiesta viene chiamato il relativo servizio BI, il quale si occuperà a sua volta di creare l'oggetto DAO corrispondente.

Descrizione dei metodi più importanti

* *public void onUpdateReceived(Update update) :* questo metodo è il metodo che viene chiamato quando arriva un qualsiasi messaggio dall'utente telegram. Il parametro update invece è il contenuto del messaggio arrivato. I messaggi interessati nel nostro caso sono quelli che contengono una stringa come testo dato che il contenuto della stringa sarà fondamentale per chiamare successivamente il servizio giusto.
* *private boolean isWhiteListed(int id):* il metodo isWhiteListed fà il controllo dell'utente se risulta registrato nella lista White dove sono memorizzati tutti gli utenti abilitati con una licenza. Il controllo viene effettuato sul parametro id dell'utente telegram. Se l'utente con codice id non fa parte della lisa White, viene mandato indietro un messaggio che avvisa l'utente di riprovare tra qualche minuto poichè si sta aspettando che un amministratore lo abiliti.
* *private void handleIncomingMessage(Message message) :* il metodo handleIncomingMessage è il gestore dei messaggi in arrivo e viene chiamato dopo il controllo del tipo di utente, ovvero se il metodo isWhiteListed ha ritornato true; handleIncomingMessage fà il paring sul contenuto del messaggio ed applica uno switch su tutti i servizi da chiamare. Questo metodo ha il compito di costruire la tastiera iniziale se il "case" corrisponde al comando di "/start", altrimenti viene chiamato uno dei servizi, il quale prende la responsabilità di costruire la tastiera in base al comando ricevuto dal contenuto dell'oggetto Message.
* *private void onWelcomeMessage(String id, String firstName, ReplyKeyboardMarkup keyboardMarkup):* questo metodo contiene il messaggio iniziale di benvenuto e si occupa di caricare il menu principale da cui scegliere i vari processi.
* *private void onGetCommand(Message message, String queryResult):* questo metodo gestisce tutti i comandi personalizzati e viene chiamato quando la query che si vuole fare risulta immediata e non necessita di un tasto nel menu.
* *private void onGetReport(Message message, String call):* onGetReport è il metodo che si occupa di inviare il report all'utente richiesto con la data e ora locale.
* *private static ReplyKeyboardMarkup getKeyboard(int idpadre):* questo metodo si occupa di creare la tastiera corrente in base all'id padre rappresentato dalla tastiera appena nascosta quando è stato premuto un bottone.

* *private static ReplyKeyboardMarkup getStartKeyboard():* questo metodo costruisce la tastiera di start con i tasti predisposti in orizzontale.

5.2.3 BI Services

In questa sottosezione vengono descritti i servizi chiamati per creare e gestire i report.

* *BDG_ Ordini_ Service:*Questo servizio si occupa di creare l'oggetto DAO che conterrà la lista degli ordini. Con un ciclo "for" si crea un unico oggetto di tipo stringa che rappresenta a secondo del DAO creato l'ordinato del giorno, l'ordinato di inizio mese oppure l'ordinato residuo. Il tipo di ritorno sarà una stringa poichè dovrà essere spedita al front-end come un messaggio di testo attraverso il metodo sendMessage(Message msg).
* *BDG_ Scadenze_ Service:*Questo servizio se occupa di creare e gestire le Scadenze che possono essere di tre tipi: Scadenze totali, Scaduto e Scadere. Anche questo servizio come BDG_Ordini_Service, in base alla richiesta dell'utente calcola e ritorna l'ammontare delle scadenze.
* *BDG_ Spedito_ Service:*BDG_Spedito_Service è uno dei tre servizi che si occupa di calcolare l'ammontare degli spediti che possono essere: Spedito oggi, Spedito ieri e Spedito questo mese. Questi tre servizi attualmente non fanno troppe operazione ma in futuro sono previsti calcoli più complessi.
* *BDG_ Disponibilità_ Service:*Questo servizio ha il compito di generare un documento con delle tabelle che contengono la disponibilità degli articoli ordinati per codice articolo. BDG_Disponibilità_Service tiene conto della profilazione per codice agente.
* *BDG_ Inevasi_ Service:*Questo servizio genera il documento degli inevasi posizionando i dati in una tabella ordinati per numero di articolo. BDG_Inevasi_Service tiene conto della profilazione per codice agente.
* *BDG_ Scadenzario_ Service:*Questo servizio genera i documenti per lo scadenzario che possono essere di tre tipi: Scaduti, a scadere e insieme Scaduti/A scadere. BDG_Scadenzario_Service tiene conto della profilazione per codice agente.

5.2.4 DAO

Lo strato DAO fondamentalmente ha una classe con relativi metodi che rappresenta un'entità tabellare di un database e viene utilizzato per stratificare e isolare l'accesso ad una tabella tramite query, che nel nostro caso sono costruite con una struttura Java o interfacciandosi direttamente con il DAO. L'accesso al data-layerG da parte della business-logicG viene quindi controllato tramite DAO, creando un maggiore livello di astrazione ed una più facile manutenibilità. I metodi del DAO con le rispettive query verranno così richiamati dalle classi della business-logic.

5.2.5 Struttura

5.3 Front-end

Capitolo 6

Conclusioni

6.1 Raggiungimento degli obiettivi

6.2 Conoscenze acquisite

6.3 Valutazione personale

Appendice A

Appendice A

Citazione

Autore della citazione

Acronimi

API Application Program Interface. 35

UML Unified Modeling Language. 35

Bibliografia

Riferimenti bibliografici

James P. Womack, Daniel T. Jones. *Lean Thinking, Second Editon*. Simon & Schuster, Inc., 2010.

Siti web consultati

Manifesto Agile. URL: http://agilemanifesto.org/iso/it/.