Università degli Studi di Padova

Dipartimento di Matematica "Tullio Levi-Civita"

Corso di Laurea in Informatica

Sviluppo di un modulo software per la gestione degli ordini di acquisto con l'utilizzo di metodi euristici di ottimizzazione

Tesi di laurea

Relatore

Prof.Luigi De Giovanni

Laureando

Filippo Brugnolaro

Matricola 1217321

Anno Accademico 2021-2022

La disumanità del computer sta nel fatto che,
una volta programmato e messo in funzione,
si comporta in maniera perfettamente onesta.

— *Isaac Asimov*

Ringraziamenti

In primis vorrei esprimere la mia gratitudine al Professor Luigi De Giovanni, relatore della mia tesi, per la disponibilità e l'aiuto fornitomi durante la stesura.

Desidero ringraziare con affetto la mia famiglia per tutto il sostegno e la vicinanza dimostrata in ogni momento e per non avermi mai fatto mancare nulla durante gli anni di studio.

Vorrei ringraziare i miei amici che mi sono stati vicini e mi hanno accompagnato in questi anni, soprattutto nei momenti difficili.

Infine desidero ringraziare in maniera speciale il mio amico Alessandro, che mi ha reso lo studio meno faticoso e con cui ho passato dei bei momenti, e Linpeng, che mi ha pazientemente guidato all'inizio del corso di laurea.

Padova, Settembre 2022

Filippo Brugnolaro

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

Nel seguente capitolo si introduce brevemente l'azienda ospitante e il progetto affrontato.

1.1 L'azienda

Ergon Informatica S.R.L.[1](da qui in poi "*Ergon*") è un'azienda italiana, fondata nel 1988, situata a Castelfranco Veneto.
Essa si occupa principalmente soluzioni gestionali per piccole e medie imprese e dello sviluppo di *software* *ERP* per i settori dell'alimentare e dei trasporti, ma completa l'offerta con la vendita di prodotti hardware, servizi *web* e *hosting*, nonché con progetti di *server consolidation* e virtualizzazione di sistemi. L'azienda inoltre si è sviluppata in maniera costante negli anni e oggi può vantare una posizione di tutto rispetto tra le aziende dello stesso settore. Attualmente fanno parte della stessa gestione:

* *Ergon Informatica S.R.L.*: che si occupa del *software*;
* *Ergon S.R.L.*: che si occupa dei servizi tecnologici;
* *Ergon Servizi S.R.L.*: che si occupa dei servizi amministrativi, logistici e di *marketing* delle altre due parti.

Il logo dell'azienda è illustrato in Figura 1.1.

Figura 1.1: Logo Ergon Informatica S.R.L.

Il prodotto proprietario dell'azienda è *ERGDIS*, sistema ERP il cui insieme dei moduli copre ogni aspetto della conduzione aziendale.

In particolare vengono gestiti vari compiti che si dislocano dall'area amministrativa al controllo direzionale, dall'area commerciale alla pianificazione e al controllo della produzione, dalla gestione acquisti alla logistica di magazzino, all'archiviazione ottica alla gestione della qualità.

[1]Sito ufficiale: https://www.ergon.it/

Alcuni di essi, inoltre, si possono interfacciare con dispositivi automatici presenti in azienda, come, ad esempio, linee di confezionamento o *robot*.

Paragrafo riguardo alle funzionalità di ERGDIS

Paragrafo riguardo agli obiettivi generali dell'azienda

1.2 L'idea

Lo *stage* proposto consiste nella progettazione e nello sviluppo di un modulo *software* volto ad assistere l'azienda nella fase di approvvigionamento dei prodotti dai propri fornitori, supportandola nel scegliere da quale fornitore e quando acquistare i prodotti.

Questa nuova funzionalità andrebbe ad ottimizzare un modulo già esistente che però, per ogni prodotto da ordinare, prende in considerazione l'ultima data d'ordine disponibile prima dell'inizio dell'effettiva copertura del fabbisogno del prodotto stesso. Questo dunque non garantirebbe con certezza una scelta ottimale in relazione alle possibilità d'ordine fornite dagli appositi listini e calendario dei fornitori.
Data la natura combinatoria del problema, il modulo dovrà fornire in tempi ragionevoli una "buona soluzione" del problema, ovvero tendente il più possibile all'ottimo, e dovrà integrarsi con l'intero sistema *ERGDIS*.

È previsto inoltre che i dati su cui si è fatta l'ottimizzazione e il confronto dei risultati vengano visualizzati tramite un'apposita interfaccia grafica che verrà sviluppata in linea con l'ambiente di sviluppo dell'azienda (*.NET Framework* e *DevExpress*).

1.3 Descrizione dello stage

1.3.1 Introduzione

1.3.2 Obiettivi

Di seguito vengono elencati tutti gli obiettivi previsti dallo *stage*:

* Analisi del contesto *ERGDIS*;
* Studio dei principali algoritmi di ricerca operativa e ottimizzazione combinatoria;
* Redazione di uno studio di fattibilità con integrazione di micro-moduli di test;
* Redazione di un'analisi dei requisiti;
* Sviluppo e codifica del modulo *software* con le tecnologie utilizzate dall'azienda;
* Redazione di documentazione tecnica riguardante le scelte implementative e architetturali effettuate;
* *Report* finale sui risultati ottenuti.

1.3.3 Analisi preventiva dei rischi

Durante la fase iniziale dello *stage*, sono stati rilevati dei possibili rischi che avrebbero potuto presentarsi durante il percorso del progetto.
Si sono dunque trovate delle soluzioni che potessero arginare i problemi. In particolare:

1. **Comprensione e confronto degli algoritmi**

   Problema: il progetto richiede un'ampia fase di studio che riguarda principalmente

la teoria di tecniche per la risoluzione di problemi di ricerca operativa e ottimizzazione combinatoria. Questo poteva portare a presentarsi la possibilità di non comprendere fino in fondo l'algoritmo, e poteva essere difficile cogliere e confrontare i pregi e difetti di ciascuno di essi.

Soluzione: è stato organizzato un incontro iniziale con il tutor per fornire una base da cui poi iniziare una ricerca più approfondita. Sono stati forniti anche delle dispense utili per rafforzare la base di partenza.

2. **Tecnologie e ambiente di sviluppo**

   Problema: venivano richieste alcune tecnologie, come per esempio *Entity Framework* o *DevExpress* a me assolutamente ignote. Sebbene avessi delle basi abbastanza solide di *C#* derivanti dalla conoscenza di altri linguaggi quali *C++* e *Java*, venivano richieste tuttavia alcune tecnologie integrate nel linguaggio (*LINQ*) anch'esse ignote. L'ambiente di sviluppo e l'*IDE* non erano mai stati utilizzati.

   Soluzione: sono stati forniti dei riferimenti consigliati per l'autoapprendimento. Tuttavia qualsiasi dubbio ragionevolmente particolare poteva essere richiesto al tutor. È stato effettuato insieme al tutor il *setup* dell'ambiente di sviluppo e la conseguente creazione dei *database*.

3. **Calibrazione dei parametri e funzione obiettivo**

   Problema: dopo la scelta e l'implementazione dell'algoritmo, è molto importante:

   * definire una funzione obiettivo che vada a descrivere in maniera "buona" l'andamento dell'algoritmo stesso;
   * calibrare i parametri in base allo spazio delle soluzioni del problema preso in esame.

   Tuttavia entrambe sono azioni molto delicate che possono compromettere il funzionamento stesso dell'algoritmo anche se implementato correttamente.

   Soluzione: cercare una costruzione e calibrazione per passi ed empiricamente dimostrabili e presentarle in una discussione con il tutor, in modo tale da creare una *baseline* su cui basarsi per continuare con i passi successivi.

1.4 Organizzazione del testo

Di seguito viene illustrata l'organizzazione dei capitoli successivi:

Il secondo capitolo approfondice lo studio di fattibilità effettuato, utile per entrare a conoscenza delle più utilizzate tecniche di ottimizzazione combinatoria e per analizzare quali siano i vantaggi e svantaggi di ognuno di essi.

Il terzo capitolo descrive l'analisi dei requisiti del progetto, comprensiva di diagrammi dei casi d'uso e raccolta dei requisiti derivanti dall'analisi di questi ultimi.

Il quarto capitolo approfondisce le fasi di progettazione e codifica, comprensiva di diagrammi delle classi e di approfondimenti a livello implementativo.

Il quinto capitolo espone tutte verifiche effettuate durante il progetto e la validazione finale a conferma dei requisiti inizialmente stilati nella fase di analisi dei requisiti.

Il sesto capitolo presenta le conclusione tratte dallo *stage*, comprensivo di conoscenze acquisite e considerazioni di carattere personale.

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;
* i termini in lingua straniera o facenti parti del gergo tecnico sono evidenziati con il carattere *corsivo*.

Capitolo 2

Studio di fattibilità

Brevissima introduzione al capitolo

2.1 Introduzione allo studio

2.2 Soluzioni proposte

2.2.1 Algoritmo Greedy

2.2.2 Algoritmo Tabu Search

2.2.3 Algoritmo Genetico

2.3 Conclusioni dello studio

Capitolo 3

Analisi dei requisiti

Breve introduzione al capitolo

3.1 Casi d'uso

Per lo studio dei casi di utilizzo del prodotto sono stati creati dei diagrammi. I diagrammi dei casi d'uso (in inglese *Use Case Diagram*) sono diagrammi di tipo UML dedicati alla descrizione delle funzioni o servizi offerti da un sistema, così come sono percepiti e utilizzati dagli attori che interagiscono col sistema stesso. Essendo il progetto finalizzato alla creazione di un tool per l'automazione di un processo, le interazioni da parte dell'utilizzatore devono essere ovviamente ridotte allo stretto necessario. Per questo motivo i diagrammi d'uso risultano semplici e in numero ridotto.

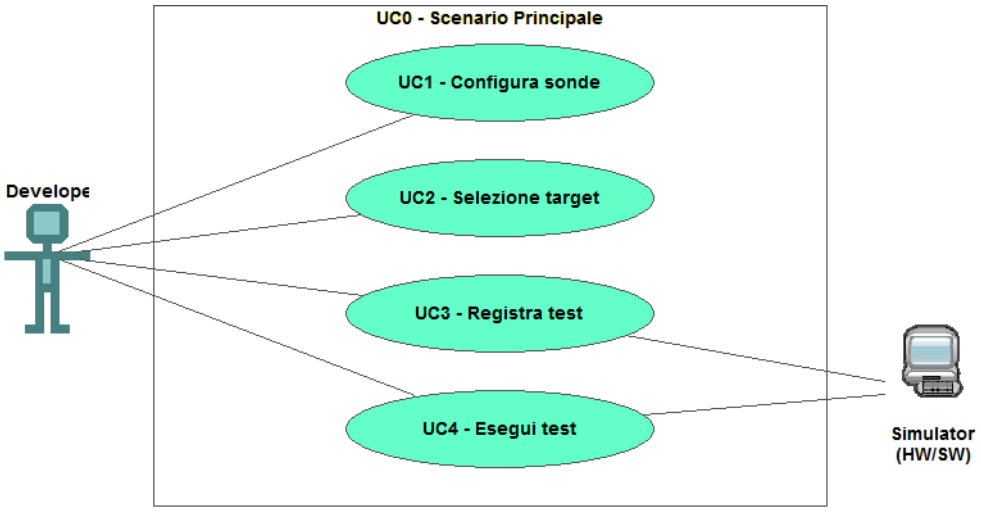

Figura 3.1: Use Case - UC0: Scenario principale

UC0: Scenario principale

Attori principali: Sviluppatore applicativi.

Pre-condizione: Lo sviluppatore è entrato nel plug-in di simulazione all'interno dell'IDE.

Descrizione: La finestra di simulazione mette a disposizione i comandi per configurare, registrare o eseguire un test.

Post-condizione: Il sistema è pronto per permettere una nuova interazione.

3.2 Tracciamento dei requisiti

Da un'attenta analisi dei requisiti e degli use case effettuata sul progetto è stata stilata la tabella che traccia i requisiti in rapporto agli use case.
Sono stati individuati diversi tipi di requisiti e si è quindi fatto utilizzo di un codice identificativo per distinguerli.
Il codice dei requisiti è così strutturato R(F/Q/V)(N/D/O) dove:

R = requisito

F = funzionale

Q = qualitativo

V = di vincolo

N = obbligatorio (necessario)

D = desiderabile

Z = opzionale

Nelle tabelle 3.1, 3.2 e 3.3 sono riassunti i requisiti e il loro tracciamento con gli use case delineati in fase di analisi.

Tabella 3.1: Tabella del tracciamento dei requisti funzionali

Requisito	Descrizione	Use Case
RFN-1	L'interfaccia permette di configurare il tipo di sonde del test	UC1

Tabella 3.2: Tabella del tracciamento dei requisiti qualitativi

Requisito	Descrizione	Use Case
RQD-1	Le prestazioni del simulatore hardware deve garantire la giusta esecuzione dei test e non la generazione di falsi negativi	-

Tabella 3.3: Tabella del tracciamento dei requisiti di vincolo

Requisito	Descrizione	Use Case
RVO-1	La libreria per l'esecuzione dei test automatici deve essere riutilizzabile	-

Capitolo 4

Progettazione e codifica

Breve introduzione al capitolo

4.1 Tecnologie e strumenti

Di seguito viene data una panoramica delle tecnologie e strumenti utilizzati.

Tecnologia 1

Descrizione Tecnologia 1.

Tecnologia 2

Descrizione Tecnologia 2

4.2 Progettazione

Namespace 1

Descrizione namespace 1.

Classe 1: Descrizione classe 1

Classe 2: Descrizione classe 2

4.3 Design Pattern utilizzati

4.4 Codifica

Capitolo 5

Verifica e validazione

5.1 Verifica

5.1.1 Documentazione

5.1.2 Testing del modulo

5.2 Validazione

5.2.1 Documentazione

5.2.2 Codice

Capitolo 6

Conclusioni

6.1 Prodotto finale

6.2 Raggiungimento degli obiettivi

6.3 Conoscenze acquisite

Appendice A

Appendice A

Citazione

Autore della citazione

Glossario

.NET Framework Ambiente di esecuzione runtime della piattaforma tecnologica .NET in cui vengono gestite le applicazioni destinate allo stesso .NET Framework. Disponibile solo su *Windows*. 2

DevExpress Framework utile per lo sviluppo di applicazioni desktop. 2, 3

ERP Tipologia di software che integra tutti i processi di business rilevanti di un'azienda e tutte le funzioni aziendali, ad esempio vendite, acquisti, gestione magazzino, finanza, contabilità... 1

Server Consolidation È un approccio all'utilizzo efficiente delle risorse dei server dei computer al fine di ridurre il numero totale di server o posizioni di server richiesti da un'organizzazione. 1

UML in ingegneria del software *UML, Unified Modeling Language* (ing. linguaggio di modellazione unificato) è un linguaggio di modellazione e specifica basato sul paradigma object-oriented. L'*UML* svolge un'importantissima funzione di "lingua franca" nella comunità della progettazione e programmazione a oggetti. Gran parte della letteratura di settore usa tale linguaggio per descrivere soluzioni analitiche e progettuali in modo sintetico e comprensibile a un vasto pubblico. 7

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/.