

ALMA MATER STUDIORUM · UNIVERSITÀ DI BOLOGNA

FACOLTÀ DI SCIENZE MATEMATICHE, FISICHE E NATURALI
Corso di Laurea in Scienze di Internet

UN SISTEMA
DI
GESTIONE DATI PER IL TRADING

Tesi di Laurea in Programmazione Internet

Relatore:
Chiar.mo Prof.
Antonio Messina

Presentata da:
Andi Drabo

II
2011/2012

Indice

Introduzione	7
1 Le basi di dati ed il trading	9
1.1 I database utilizzati nel trading	9
1.2 Il Database PostgreSQL	10
1.2.1 Descrizione	11
1.2.2 Funzionalità	12
1.2.3 La struttura dei dati	13
2 Architettura del Sistema	17
2.1 Specifica dei Requisiti	17
2.2 La struttura del database	18
2.3 Le Funzionalità	23
2.4 Diagrammi UML	25
2.4.1 Descrizione dell'UML	25
2.4.2 Diagramma di Attività	27
2.4.3 Diagramma dei Casi d'uso	29
2.4.4 Diagramma di Sequenza	32
2.4.5 Diagramma di Componenti	36
2.4.6 Diagramma delle Classi	37
2.4.7 Diagramma di Deployment	41
3 Fasi di Sviluppo	45
3.1 Progettazione	45
3.2 Implementazione	45
3.2.1 Descrizione delle Classi	46
3.3 Prima Fase di Test	47
3.4 Debugging	48
3.5 Seconda Fase di Test	48

4 Realizzazione	51
4.1 Descrizione dell'applicativo	51
4.2 Misure di prestazione	63
Conclusioni	67
Bibliografia	69
Elenco delle figure	71

Introduzione

L'utilizzo e lo sviluppo sempre più crescente della rete Internet rappresenta, sia per totale e capillare diffusione, sia per la facilità di accesso a costi contenuti, una straordinaria opportunità di crescita per i mercati finanziari, tanto da essere divenire un vero e proprio ausilio tecnico per l'offerta di erogazione di servizi d'investimento. Tramite la rete, inoltre, risulta molto più semplice accedere ai mercati esteri, avvantaggiando coloro che intendono effettuare investimenti. Oltre ai limiti di spazio, un'altra barriera che lo strumento informatico permette di superare è quella temporale: la velocità del mezzo è un elemento caratterizzante di ogni comunicazione informatica. Il trading[1, 2, 3, 4] on line rappresenta l'area principale dell'Internet banking, dal momento che l'immaterialità del denaro e del concetto stesso di possesso consentono di rendere il mondo della rete ideale per questo tipo di transazioni. La tipologia dei servizi che gli intermediari finanziari offrono tramite Internet è molto ampia, e si estende dai servizi di informazione, a quelli di utilità e di interattività. Il primo gruppo è costituito dalle informazioni che l'intermediario fornisce sul proprio conto e sui servizi offerti, il secondo rappresenta un tentativo di fidelizzazione del cliente tramite informazioni culturali, economico-finanziarie, commerciali, ed il terzo gruppo rappresenta la vera frontiera dell'Internet banking, in cui si collocano i servizi informativi e quelli dispositivi. È proprio all'interno di questa categoria che si inserisce il trading on line. Il fenomeno nasce negli Stati Uniti nei primi anni '80, per ovviare alle esigenze di grandi investitori americani. La prima collocazione di prodotti finanziari per mezzo della rete ha avuto luogo nel 1996 ad opera di una piccola società newyorkese produttrice di birra, che riuscì a raccogliere in breve tempo, sottoscrizioni per un totale di 1.600.000 dollari tra circa 3.500 investitori, senza fare ricorso ad alcun intermediario. Per fare ciò la società creò semplicemente una pagina web dalla quale gli investitori potevano scaricare i documenti contenenti i termini dell'offerta pubblica di sottoscrizione delle azioni di nuova emissione.

In Italia la prima società ad operare nel settore è stata nel 1996 Di-

recta SIM, ma solo dopo il 1998 il trading on line ha iniziato a suscitare un significativo interesse anche nel nostro paese. Oggi, complici anche una Borsa estremamente effervescente ed un calo sempre più marcato nei rendimenti degli strumenti di investimento tradizionali (BOT in testa), nessuna istituzione finanziaria medio-grande può più permettersi di prescindere da un'offerta di trading on line. Negli ultimi anni, con l'evoluzione delle tecnologie informatiche sono sempre di più le persone coinvolte nel trading online.

Il termine trading indica la compravendita on line di azioni o altri beni finanziari. Detto diversamente il trading on line indica la negoziazione di valori mobiliari realizzata attraverso il proprio computer. I vantaggi del trading on line per il privato forniscono sia la possibilità di operare su quasi tutti i mercati mondiali direttamente dal proprio ufficio o da casa, sia ottenere costi sicuramente inferiori rispetto a quelli legati alle operazioni in rete tradizionali. Questo perchè le commissioni di intermediazione e di tenuta del conto sono relativamente più basse, dal momento che gli ordini di acquisto e di vendita vengono immessi dal cliente ed eseguiti tramite sistemi telematici, evitando così ulteriori commissioni che incrementano i costi. Inoltre, come detto in precedenza, esiste un evidente risparmio di tempo e, soprattutto, tempestività delle operazioni, le quali possono essere eseguite in tempo reale, abbassando al minimo il rischio derivato dalla possibilità di variazioni di prezzo tra il momento in cui viene presa la decisione e la reale esecuzione dell'ordine. Infine, la grande disponibilità di informazioni, pone gli investitori e gli speculatori nelle stesse condizioni di effettuare le proprie scelte con le medesime modalità di trattamento.

Tra i problemi più frequenti a cui si può andare incontro nell'operare nel trading on line, emergono quelli tecnici, come problemi legati al server delle banche a cui si fa riferimento. Mentre si sta immettendo un ordine si può ricevere un messaggio di errore di collegamento, che impedisce di immettere ordini o di eseguire altre operazioni. Sono molti gli strumenti ed i programmi disponibili al giorno d'oggi per fare trading online. I dati giocano un ruolo fondamentale in questo tipo di attività. I dati del mercato sono disponibili in grandi quantità e possono coprire amplissimi intervalli di tempo. Per memorizzare questi dati, elaborarli, studiarli e usarli correttamente nel processo del trading un programma informatico può essere molto utile. Scopo di questa tesi è quello di progettare e implementare un sistema software proprio per la gestione di questi dati. Il sistema deve avere delle caratteristiche specifiche e deve soddisfare delle richieste sulla qualità. In particolare, deve essere: facile da utilizzare, affidabile e veloce. Il sistema che è stato realizzato è composto da un database nel quale vengono

memorizzati i dati e dalla parte software che elabora tali dati e gestisce il database stesso. In particolare, nel primo capitolo di questa tesi vengono descritti i modelli di database utilizzati nel trading e spiegato in dettaglio il specifico database usato in questo progetto. Nel secondo capitolo viene illustrata l'architettura del sistema creato insieme ad una dettagliata descrizione delle funzionalità. Il terzo capitolo delinea le fasi di sviluppo del software che prevedono, in linea di massima la progettazione, l'implementazione e la fase di test. Infine, nel quarto capitolo viene descritta la realizzazione dell'applicativo.

Capitolo 1

Le basi di dati ed il trading

Fare trading significa lavorare con grandi quantità di dati. Questo perché i trader fanno operazioni con dati che coprono grandi archi temporali. Si pensi ad esempio ad una grande multinazionale che da più di venti anni sia quotata in borsa. Ognuna delle borse, in cui l'azione è quotata, può fornire di questa azione, per ogni giorno di apertura:

- il prezzo di apertura;
- il prezzo di chiusura;
- il prezzo più alto fra l'apertura e la chiusura;
- il prezzo più basso fra l'apertura e la chiusura;
- la quantità di azioni trattate;
- il prezzo fra quello di apertura e quello di chiusura, ad intervalli prefissati.

La quantità dei dati disponibili in questo caso sarà ovviamente enorme. Abbiamo bisogno di memorizzare questi dati, di manipolarli, di memorizzare i risultati ottenuti, di studiare il loro andamento, di aggiungere commenti di diverso tipo quali commenti audio, file di testo oppure grafici. Per fare tutto ciò in maniera sicura, facile, affidabile e veloce abbiamo bisogno di una base di dati.

1.1 I database utilizzati nel trading

Come tutti sanno esistono diversi tipi di sistemi di gestione di basi di dati. Questi sistemi hanno caratteristiche e funzionalità diverse l'uno con l'altro.

Viene naturale porsi la domanda quale tra tutti questi sistemi è più adatto per essere integrato in un software utilizzato per il trading. Per rispondere a questa domanda bisogna tenere conto di diversi fattori quali la struttura dei dati, le relazioni che possiamo creare con questi dati una volta caricati in database e dei tipi di dati nuovi che vogliamo creare come ad esempio commenti vocali o grafici.

Il modello che meglio si adatta ai sistemi software di trading è sicuramente il modello relazionale. Esso si basa sulla teoria degli insiemi e sulla logica del primo ordine ed è strutturato attorno al concetto di relazione (realizzata in una tabella). Inoltre consente al progettista di database di creare una rappresentazione consistente e logica dell'informazione.

La struttura base del modello relazionale è il dominio o tipo di dato, definito come l'insieme dei valori che può assumere un determinato attributo. Una tupla è un insieme non ordinato di valori degli attributi. Un attributo è una coppia ordinata di 'nome di attributo' e 'nome di tipo', mentre un valore di attributo è un valore specifico valido per quel tipo di dato. Una relazione consiste di una testata e di un corpo, dove la testata è un insieme di attributi e il corpo è un insieme di n tuple. La testata di una relazione è anche la testata di ciascuna delle sue tuple. La tabella è la rappresentazione grafica normalmente accettata per rappresentare la relazione.

Il principio base del modello relazionale è che tutte le informazioni siano rappresentate da valori inseriti in relazioni (tabelle); dunque un database relazionale è un insieme di relazioni contenenti valori e il risultato di qualunque interrogazione (o manipolazione) dei dati può essere rappresentato anch'esso da relazioni (tabelle).

1.2 Il Database PostgreSQL

PostgreSQL[5] è un potente sistema di database relazionale open source. Il suo sviluppo e miglioramento è da più di 15 anni che va avanti. La sua architettura ha guadagnato una collaudata reputazione per la sua affidabilità e per la correttezza ed integrità dei dati. Funziona su tutti i principali sistemi operativi, inclusi Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64), e Windows.

È pienamente ACID¹ compatibile, ha pieno supporto per le chiavi esterne, join, viste, trigger e stored procedure (in più lingue). Comprende la maggior caratteristiche di un linguaggio SQL: 2008 tipi di dati, tra cui

¹ACID, Atomic, Consistent, Isolated, Durable

INTEGER, NUMERIC, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL e TIMESTAMP. Supporta inoltre lo stoccaggio di oggetti binari di grandi dimensioni, tra cui immagini, suoni o video. Dispone di interfacce di programmazione nativi per C / C ++, Java, . Net, Perl, Python, Ruby, Tcl, ODBC.

Di seguito verranno spiegate in maniera più approfondita le sue funzionalità ed anche una breve storia di questo sistema di basi di dati.

1.2.1 Descrizione

Date le sue caratteristiche potenti e avanzate, ci si chiede come un prezioso sistema software possa essere open source. Come per molti altri progetti importanti open source, lo sviluppo del sistema inizia presso l'Università della California a Berkeley (UCB).

PostgreSQL, originariamente chiamato Postgres, è stato creato alla Università di California Berkeley da un professore di informatica di nome Michael Stonebraker, che poi è diventato il Chief Technology Officer di Informix Corporation.

Stonebraker ha iniziato a lavorare su Postgres nel 1986 basandosi sul suo predecessore, Ingres che ora è proprietà di Computer Associates. Da qui la scelta del nome, Postgres ('dopo Ingres').

Stonebraker ed i suoi studenti hanno continuato il sviluppo di Postgres per otto anni. Durante questo periodo sono state introdotte regole, procedure, tipi estensibili con indici e concetti di relazioni tra oggetti.

Postgres è stato successivamente commercializzato per diventare Illustra che fu poi acquistata da Informix e integrato nel suo Universal Server. Informix è stata acquistata da IBM nel 2001 per un miliardo di dollari. Nel 1995, due studenti provenienti dal laboratorio di Stonebraker, Andrew Yu e Jolly Chen, sostituiscono il linguaggio di interrogazione di Postgres chiamato POSTQUEL con un esteso sottoinsieme di SQL. Hanno inoltre rinominato il sistema come Postgres95.

Nel 1996 Postgres esce dal mondo accademico ed inizia una nuova vita nel mondo open source quando un gruppo di sviluppatori fuori da Berkeley, rendendosi conto delle potenzialità del sistema, decidono di continuare con lo sviluppo.

Contribuendo con enorme lavoro, tempo e capacità questo gruppo riuscì a trasformare radicalmente Postgres. Nel corso dei successivi otto anni, hanno portato coerenza e uniformità alla base di codice, creato test di regressione dettagliati per assicurarne la qualità, istituito mailing list per segnalazioni di bug, sistemato innumerevoli bug, aggiunto nuove incredibili

caratteristiche, e completato il sistema eliminando diverse lacune come la documentazione per sviluppatori e utenti.

Il risultato del loro lavoro è il nuovo database che ha ottenuto una grande reputazione per la stabilità. Con l'inizio della sua nuova vita nel mondo open source, con molte nuove funzionalità e miglioramenti, il sistema di database ha preso il suo nome attuale: PostgreSQL. ('Postgres' viene ancora usato come facile da pronunciare oppure come un semplice nomignolo).

1.2.2 Funzionalità

Le più importanti funzionalità e caratteristiche del sistema di basi di dati PostgreSQL sono:

- Sintassi ANSI SQL 89, 92 and 99;
- Transazioni ACID compatibili al 100%. Pieno supporto per il Commit e Rollback;
- Serializable / Read Committed;
- Online Backup: ottima sicurezza e disponibilità dei dati;
- Tipi di dati: numeric, decimal, smallint, integer, bigint, real, double, serial, char, varchar, bit, text, date, time, timestamp, interval, boolean, network address, geometric e tanti altri;
- Altri tipi di dati possono essere definiti dall utente;
- Memorizzazione di BLOBs (Binary Large Objects) incluso testo, audio, foto e video;
- Integrità referenziale: usa CREATE TABLE...FOREIGN KEY, cascade, restrict, set default, set null;
- Aggregazioni: pieno supporto per il GROUP BY e funzioni aggregate come COUNT, SUM, AVG, MIN, MAX, STDDEV e VARIANCE. Altre funzioni aggregate posso essere creati dall utente;
- Joins: implementa tutti i tipi: cross, inner, outer, left, right, full, natural;
- Views: supporta standart ANSI e lagggiornamento delle viste;

- Funzioni definite dall'utente: possono essere scritte in diversi linguaggi: C, SQL, PL/pgSQL (simile ad Oracle PL/SQL), TCL, Perl, Python e Ruby;
- Linguaggi di programmazione ed interfacce: Perl, Python, Zope, PHP, TCL/TK, ODBC, JDBC, C/C++, Embedded SQL, Delphi/Kylix/Pascal, VB, ASP, Java;
- Funzioni e librerie di operatori con diverse funzioni pre-installati (math, date/time, string, geometric, etc);
- Triggers: funzioni di trigger possono essere definiti usando diversi linguaggi supportati quali C o PL/PgSQL;
- Sistema di controllo della concorrenza esclusivo e multi-versione per applicazioni con grandi esigenze di concorrenza. Grazie a questo meccanismo i lettori non bloccano i scrittori e viceversa. Assenza di deadlock;
- Tabelle temporanee: eliminate automaticamente alla fine della sessione;
- Dimensione del database e tabelle virtualmente illimitata. Record ed indici illimitati per tabella;
- Modello di sicurezza standart user/group. L'accesso al server del database può essere limitato dall'utente;

1.2.3 La struttura dei dati

La struttura dei dati utilizzati dal sistema per fare operazioni di trading è di fondamentale importanza per capire il suo funzionamento. I dati che il sistema riceve, analizza, manipola, memorizza e utilizza per creare dei grafici hanno una struttura precisa e ben formattata come verrà spiegato in questa sezione.

I nostri dati rappresentano i prezzi di azioni in uno o più mercati azionari relativi ad una o più azienda quotata in tali mercati. Questi prezzi ovviamente possono essere differenti in diversi archi temporali. Non solo, i prezzi possono variare anche durante la singola giornata. Per tenere traccia di quest'ultima variazione si usa, per rappresentare i dati, un modello chiamato OHLC².

Questi valori rappresentano rispettivamente:

²OHLC sta per Open-High-Low-Close

- Open: il prezzo dell'azione nel momento dell'apertura del mercato azionario o della borsa;
- High: il prezzo massimo che l'azione assume durante la giornata;
- Low: il prezzo minimo che l'azione assume durante la giornata;
- Close: il prezzo dell'azione nel momento della chiusura del mercato o della borsa;
- Volume: il numero totale delle azioni trattate;
- AdjClose: la chiusura aggiustata.

Vediamo ora in dettaglio cosa rappresenta e come viene calcolato l'ultimo valore dell'elenco, la chiusura aggiustata. Durante il corso di una giornata di negoziazione possono accadere molte cose, capaci di influenzare il prezzo di una azione. Tale prezzo può essere cambiato, per esempio, da una qualsiasi sorte di distribuzione che viene fatta agli investitori. Le distribuzioni possono comprendere:

- Cash dividends³;
- Stock dividends;
- Stock splits.

Cash dividend indica il denaro pagato agli azionisti normalmente fuori dal guadagno corrente ed i profitti accumulati dall'azienda.

Stock dividends definisce un pagamento di dividendi effettuato sotto forma di azioni aggiuntive, piuttosto che un pagamento in contanti.

Stock splits indica una azione aziendale in cui azioni esistenti di una società sono divisi in più azioni. Lo split implica un'incremento del numero di azioni in circolazione ma senza cambiare il valore complessivo delle azioni, perché nessun valore reale è stato aggiunto a seguito della scissione. Quando vengono fatti delle distribuzioni è facile calcolare la chiusura aggiustata. Nel primo caso, quello dei cash dividends, il valore del dividendo viene sottratto dal valore dell'ultima chiusura. Assumiamo, per esempio, che il prezzo di chiusura dell'azione di una determinata azienda in uno specifico giorno sia 20\$. Il giorno successivo la stessa azienda annuncia un cash dividend di 2.50\$ per azione. La chiusura aggiustata sarà dunque: 17.50\$ (20\$

³Dividends o Dividendi sono pagamenti che un'azienda è in grado di pagare periodicamente ai propri azionisti.

- 2.50\$). Se l'azienda, invece, al posto del cash dividend annuncia uno stock dividend, p.es. 2:1⁴, il valore della chiusura aggiustata sarà $20\$ * (1/(2+1))$ che è circa pari a 6.67\$.

Infine, se l'azienda decide di effettuare una distribuzione di tipo stock split 2:1 gli investitori riceveranno una azione extra per ciascuna azione in loro possesso. In questo caso il valore della chiusura aggiustata sarà dunque $20\$ * (1/(1x2))$ che è pari a 10\$.

Vediamo ora come sono strutturati i dati relativi ai prezzi poc'anzi. Poiché questi dati sono riferiti ad uno specifico giorno, ciascuna tupla di valori OHLC deve essere associata ad un'unica data giornaliera. Un semplice esempio di questi dati viene fatto dalla figura seguente.

Data	Open	High	Low	Close	AjdClose	Volume
2000-01-01	34.21	54.24	33.09	45.10	44.12	345654
2000-01-02	30.87	35.22	28.19	29.05	30.05	345655
2000-01-03	40.00	47.20	37.06	37.04	37.02	345656

Figura 1.1: Struttura dai

La tabella contiene i dati relativi ai diversi prezzi dell'azione durante la giornata per tre giornate consecutive. Le modalità in cui i dati vengono utilizzati dal sistema verranno spiegate in dettaglio nel capitolo successivo.

⁴2:1 stock dividend significa che per ciascuna azione che l'investitore possiede ne riceverà altri 2 in più

Capitolo 2

Architettura del Sistema

2.1 Specifica dei Requisiti

Un punto cruciale per la realizzazione di un sistema software è specificare dettagliatamente quello che il sistema stesso dovrà fare. Senza un'adeguata comprensione del dominio è difficile identificare le interazioni che il sistema software avrà nello svolgimento delle sue funzionalità. Questo può comportare delle grosse difficoltà in fase di progettazione ed in fase di implementazione.

Proprio per ovviare a questi problemi è stato dedicato particolare attenzione a questa fase. Il sistema dovrà gestire grandi quantità di dati, dovrà poter importare i dati sia da un file CSV, sia in modo automatico. Su questi dati, in un primo momento verranno fatti dei controlli (correttezza formato, gestione duplicati etc.) per essere poi inseriti in una apposita tabella nel database.

L'utente potrà visualizzare i dati inseriti secondo diversi criteri di ricerca. Inoltre dovrà essere informato tramite messaggi di errore nel caso in cui i criteri scelti siano errati. Le operazioni che l'utente potrà effettuare oltre alle semplici manipolazione dati comprendono operazioni di statistica, creazione/memorizzazione/cancellazione di grafici e registrazione e ascolto di commenti vocali. L'elenco sottostante illustra in dettaglio i requisiti del sistema. L'utente potrà:

- Connettersi al database inserendo il proprio username e password;
- Visualizzare i dati disponibili secondo diversi criteri quali il nome dell'azione, l'intervallo temporale etc.;
- Inserire nuovi dati in database importandoli da un file CSV oppure in modo automatico dal sito finance.yahoo.com;

- Cancellare porzione dei dati;
- Fare operazioni di statistica scegliendo se memorizzare i risultati oppure scartarli;
- Ricercare in modo facile i risultati salvati e visualizzarli;
- Creare grafici 2D in una o più variabili;
- Creare grafici OHLC;
- Memorizzare i grafici oppure scartarli;
- Visualizzare i grafici memorizzati;
- Registrare commenti vocali;
- Ascoltare i commenti registrati.

2.2 La struttura del database

In questa sezione viene descritto il modello del database utilizzato per la realizzazione dell'applicativo. Un punto cruciale dell'intera fase di sviluppo è stato proprio la progettazione della base di dati. Lo strumento utilizzato per creare il modello della base di dati è Workbench[6], uno strumento grafico completo per progettare, modellare, generare e gestire visivamente database ed offre funzionalità di data modeling e sviluppo SQL. Permette, inoltre, di progettare database basati su modelli, creando quindi, in modo efficiente, database solidi e ben funzionanti, garantendo così flessibilità necessaria per far fronte alle esigenze dei progettisti. Di seguito viene mostrato lo schema relativo alla base di dati, seguita da una spiegazione dettagliata delle tabelle e delle rispettive relazioni.

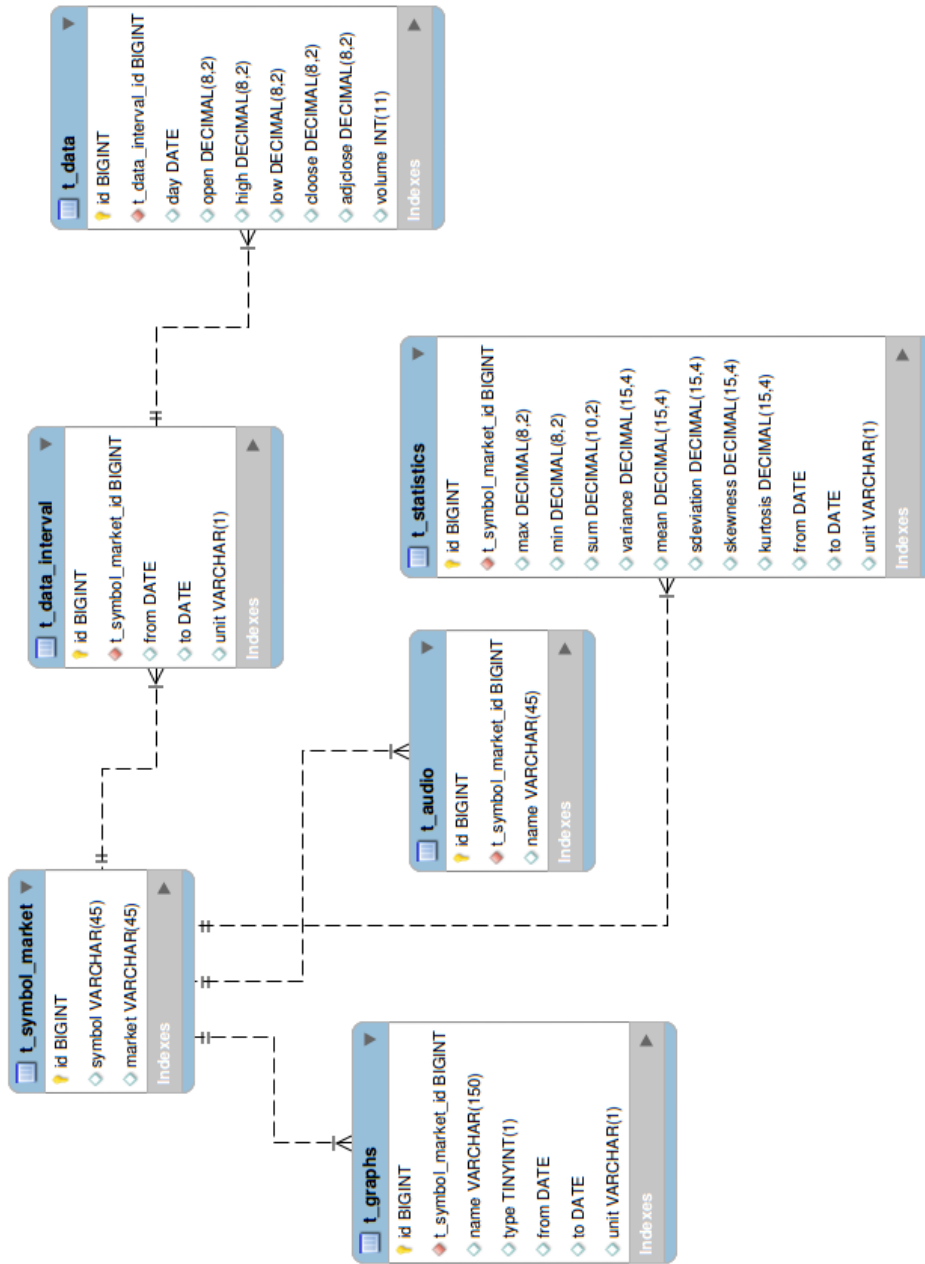


Figura 2.1: Struttura del database

Il database è composto da 7 tabelle: *t_symbol_market*, *t_data_interval*, *t_data*, *t_graphs*, *t_audio* e *t_statistics*. La prima tabella, *t_symbol_market* è composta da 3 campi: “id”, “symbol” e “market”. L’“id” è un numero intero definito come BIGINT e auto-increment e identifica in modo univoco ciascuna riga all’interno della tabella stessa. Questo campo è presente in tutte le tabelle ed in ognuna di esse individua la riga corrispondente. I campi “symbol” e “market”, definiti entrambi come VARCHAR(45), contengono rispettivamente il nome dell’azione e quello del mercato a cui si riferisce. La tabella *t_data_interval* contiene le informazioni relative ai dati disponibili nel database. È composta da 5 campi: “id”, “t_symbol_market”, “from”, “to”, “unit”. Il campo “t_symbol_market” è definito come una chiave esterna che fa riferimento al campo “id” della tabella *t_symbol_market*. Il campo “from”, definito come DATE, contiene il giorno d’inizio dell’intervallo di tempo di cui si dispongono i dati relativi alla riga della tabella *t_symbol_market*. Analogamente il campo “to” contiene il giorno di fine dell’intervallo temporale. Infine, il campo “unit”, definito come VARCHAR(1) viene usato per indicare qual’è la frequenza dei dati. I valori che esso può assumere sono i 3 caratteri: “d” (frequenza giornaliera), “w” (frequenza settimanale), “m” (frequenza mensile). Questa tabella viene utilizzata principalmente per effettuare delle ricerche veloci sui dati disponibili su un determinato intervallo di tempo e una determinata frequenza.

Nella tabella *t_data* vengono memorizzati i dati OHLC. Essa è composta da 9 campi: “id”, “t_data_interval_id”, “day”, “open”, “high”, “low”, “close”, “adjclose” e “volume”. Il campo “t_data_interval_id” è una chiave esterna collegata con la chiave primaria della tabella *t_data_interval*. Il campo “day” contiene il giorno a cui si riferiscono i valori “open”, “high”, “low”, “close”, “adjclose” e “volume”. Questi sono definiti come DECIMAL(8,2) cioè come numeri decimali a 8 cifre, di cui 2 cifre dopo la virgola. Tali campi contengono rispettivamente il valore dell’azione al momento dell’apertura, il valore massimo e minimo della giornata, il valore al momento della chiusura, il valore della chiusura aggiustato ed il numero totale delle azioni trattate.

La tabella *t_graphs* contiene informazioni relative ai grafici memorizzati in database. Essa è composta da 7 campi: “id”, “t_symbol_market_id”, “name”, “type”, “from”, “to”, “unit”. Il campo “t_symbol_market_id” è una chiave esterna alla principale “t_symbol_market”. Il campo “name”, definito come VARCHAR(150), contiene il nome del grafico che viene scelto dall’utente. Il campo “type” definito come TINYINT(1) viene usato per indicare il tipo del grafico. I grafici standard ad una variabile vengono identificati con il valore 1, i grafici standard a più variabili con il valore 2

ed i grafici OHLC con il valore 3. I campi “from” e “to”, entrambi definiti come DATE contengono il giorno di inizio ed il giorno di fine dell’intervallo di tempo su cui il grafico è stato creato. Infine, il campo “unit” viene usato per contenere la frequenza temporale dei dati che sono stati presi in considerazione per la creazione del grafico.

La tabella *t_statistics* viene utilizzata per memorizzare i risultati ottenuti tramite elaborazioni statistiche. Contiene un campo per ciascuna operazione statistica che il sistema può effettuare, insieme ai tre campi “from”, “to” e “unit” i cui ruoli sono identici alla tabella *t_graphs*.

Infine, la tabella *t_audio* contiene le informazioni relativi ai commenti vocali registrati dall’utente e presenti nel sistema. Essa contiene una chiave esterna alla tabella *t_symbol_market* ed un campo “nome” definito come VARCHAR(45) che contiene il nome del commento vocale. La base di dati progettata è il cuore del sistema. In essa vengono memorizzati tutti i dati OHLC insieme agli risultati statistici.

Per i grafici ed i commenti vocali si è scelto di memorizzarli sul disco, invece che nel database, e inserire nelle apposite tabelle soltanto l’informazione necessaria per recuperare questi oggetti nel futuro. La ragione primaria di questa scelta è che in questo modo si risparmia notevolmente memoria e si hanno migliori prestazioni. Diversi studi[7] hanno dimostrato che per oggetti (file) di dimensioni fino a 256K il database risulta avere delle prestazioni migliori del filesystem. Invece, se la dimensione è maggiore di 256K è il filesystem che risulta essere più performante del database. Uno dei principali problemi che può derivare dalla memorizzazione di file di grandi dimensioni in database è la frammentazione dei dati. I moderni filesystem sono molto meno influenzati da questo problema rispetto ai sistemi di database. L’applicativo salva i grafici ed i commenti vocali sul disco fisico in diverse cartelle che sono organizzate secondo una specifica gerarchia e con determinati formati. I grafici vengono salvati come file con estensione JPG mentre i commenti vocali con estensione MP3. Il modo su cui vengono strutturate le cartelle viene descritto dalla figura seguente:

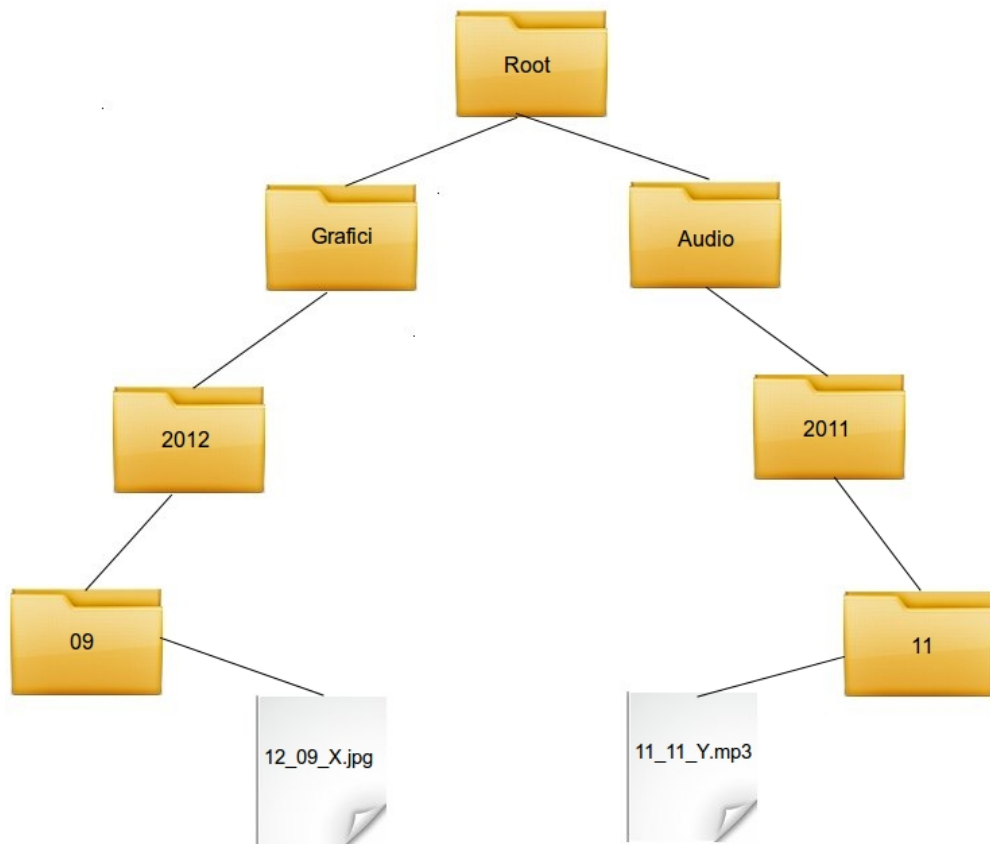


Figura 2.2: Gerarchia cartelle

La cartella Root costituisce la radice della struttura. Dentro questa cartella vengono create le 2 sottocartelle, Grafici e Audio. La prima contiene i file immagine relativi ai grafici creati mentre la seconda i file audio registrati. All'interno di ciascuna di loro vengono creati delle sottocartelle che hanno come nome l'anno (o l'intervallo di anni) in relazione al quale (o ai quali) sono stati creati i grafici oppure registrati i commenti. Ogni cartella "anno" contiene all'interno delle sottocartelle "mese" che analogamente hanno come nome il numero del mese corrispondente ai file a loro interno. I file vengono salvati proprio dentro queste cartelle. Il nome dei file viene costruito, come si vede dalla figura, concatenando le ultime 2 cifre dell'anno con il numero del mese e una stringa che l'utente può scegliere. Quest'ultima informazione rappresenta il nome vero e proprio del file, ed è questo nome che viene inserito nell'apposita tabella nel database. Questo approccio permette al sistema di poter recuperare i file in maniera veloce e sicura ogni volta che l'utente lo richiede. Il primo controllo viene fatto sul nome del file e sul tipo. Se il nome file esiste in database il sistema lo recupera cercando all'interno dell'albero delle cartelle, procedendo per anno e per mese. In base al tipo del file che l'utente vuole visualizzare/ascoltare, il sistema sceglie l'azione appropriata da compiere.

2.3 Le Funzionalità

Il software è stato ideato per la gestione di dati in un database per il trading. Il sistema è dotato di un 'motore di ricerca' in grado di recuperare e visualizzare in maniera efficiente tutta l'informazione contenuta nel database. Per l'importazione dei dati direttamente dal sito <http://finance.yahoo.com> viene utilizzata una libreria chiamata ojalgo[8], che attraverso le API di Yahoo! permette appunto di scaricare sul calcolatore locale i dati OHLC. Una volta caricati i dati forniti dal sito, l'applicativo software consente di navigare nella banca dati utilizzando funzioni di ricerca che possono essere effettuate attraverso diversi parametri come, per esempio, il nome dell'azione, l'intervallo di data ed il mercato.

Sono disponibili, inoltre, elaborazioni statistiche sui gruppi di record selezionati, che consentono, per esempio, di calcolare la media, la somma, la varianza, la media, la skewness etc. Il sistema, offre, dunque, la possibilità all'utente di gestire in maniera ottimale le principali funzioni della gestione di dati.

Attraverso un'interfaccia grafica molto intuitiva, l'utente può facilmente navigare nel programma ed effettuare le operazioni desiderate. Il sistema è diviso in 4 sottosistemi che sono:

1. Gestione dei dati numerici e non;
2. Creazione di grafici di diversi tipi;
3. Elaborazioni statistiche;
4. Registrazione di commenti vocali.

Vediamo ora più in dettaglio le funzionalità di queste parti considerandole una ad una. Il sottosistema che gestisce i dati si occupa delle seguenti attività:

- Controllo stato database;
- Creazione delle tabelle;
- Gestione richiesta di importazione dati;
- Controllo della modalità di importazione;
- Setacciamento dei dati;
- Controllo coerenza dati (dati mancanti, dati duplicati, intervalli errati etc.);
- Inserimento dati in database che comporta anche un aggiuntivo controllo del database stesso;
- Selezione record nel database;
- Produzione di opportuni messaggi di errore in caso di parametri non corretti in fase di visualizzazione.

L'elenco dettagliato delle funzionalità del sottosistema che gestisce la creazione dei grafici viene descritto di seguito:

- Identificazione del tipo di grafico;
- Identificazione dell'intervallo di tempo;
- Identificazione del numero di variabili su cui si vuole fare il plot;
- Creazione del grafico e visualizzazione dello stesso;
- Possibilità all'utente di manipolare il grafico tramite numerose opzioni di visualizzazione;

- Controllo coerenza dati (dati mancanti, dati duplicati, intervalli errati, etc.).

Il modulo che gestisce la parte della statistica viene descritto, in termini di funzionalità dall'elenco di seguito:

- Identificazione dell'operazione o delle operazioni da effettuare;
- Possibilità di fare più operazioni e su più record di dati separati;
- Interfaccia con elevato grado di usabilità sia per fare le richieste che per vedere i risultati prodotti;
- Produrre significativi messaggi di warning nel caso l'utente voglia calcolare risultati già presenti nel database.

Infine, il sottosistema responsabile della registrazione e la riproduzione dei commenti audio offre le seguenti funzionalità:

- Identificazione del tipo di commento (relativo ai dati numerici, a grafici, previsioni future ecc.);
- Registrazione del commento vocale;
- Riproduzione dei commenti vocali.

2.4 Diagrammi UML

2.4.1 Descrizione dell'UML

L'UML¹ è un linguaggio visuale di modellazione di sistemi software. Tale linguaggio deriva dall'integrazione di modelli preesistenti come Booch, OMT² e OOSE³ ed i suoi autori sono Booch, Rumbaugh e Jacobson, che si riunirono alla Rational Corporation per lavorare sull'UML. Nel 1996 l'OMG⁴ produsse una RFP⁵ per un linguaggio di modellazione visuale, a cui venne sottoposto l'UML. Nel 1997 l'OMG approvò l'UML e nacque così il primo standard non proprietario dell'industria per un linguaggio di modellazione visuale. La premessa fondamentale dell'UML è la possibilità

¹Unified Modeling Language

²Object-modeling technique

³Object-oriented software engineering

⁴Object Management Group

⁵Request For Proposal

di modellare sistemi software come insieme di oggetti che collaborano tra loro. Tale metodologia è molto adatta a sistemi software e a linguaggi OO⁶, ma si presta anche molto bene per modellare i processi aziendali ed altri settori applicativi. Un modello UML ha 2 aspetti che riguardano una struttura statica che descrive i tipi di oggetto necessari per modellare il sistema ed un comportamento dinamico che invece illustra il ciclo di vita di questi oggetti ed il loro modo di collaborare per fornire le funzionalità richieste dal sistema. Questi 2 aspetti sono complementari. Per capire come l'UML possa funzionare come linguaggio visuale possiamo esaminarne la struttura che è composta da:

- Entità: elementi di modellazione;
- Relazioni: legano tra loro le entità e definiscono come le varie entità sono semanticamente correlate;
- Diagrammi: mostrano l'insieme degli elementi che descrivono il sistema software e visualizzano cosa farà il sistema.

Le entità, in particolare possono essere di tipo strutturale (classe, interfaccia, caso d'uso, etc.), comportamentale (interazioni), di raggruppamento (package) e di tipo informativo (necessarie per fornire informazioni particolari). Le relazioni invece, mostrano come due o più entità sono correlate tra loro e consentono di fissare i legami semantici tra gli elementi. Esse si possono applicare alle entità strutturali e di raggruppamento. I diagrammi sono *viste* o *finestre* che consentono di vedere il contenuto del modello. Esistono 13 diversi tipi di diagrammi UML tra cui i "diagrammi di attività", dei "casi d'uso", di "sequenza", dei "componenti" di "deployment" e delle "classi" che verranno esaminati dettagliatamente di seguito.

Un sistema software grazie al linguaggio UML viene disegnato professionalmente e documentato ancor prima che ne venga scritto il relativo codice, da parte degli sviluppatori. Questo permette di conoscere in anticipo il risultato finale del progetto su cui si sta lavorando e poichè, la fase di disegno del sistema precede la fase di scrittura del codice, ne consegue che la scrittura del codice stesso è resa più agevole. L'UML, inoltre, essendo un linguaggio di modellazione per un sistema software consente di poter riutilizzare parte del codice abbassando così anche i costi di sviluppo di un progetto.

L'utilizzo dei diagrammi UML permette di avere una chiara idea, a chiunque sia coinvolto nello sviluppo, di tutto l'insieme che costituisce il sistema. In questo modo, si potranno sfruttare al meglio anche le risorse

⁶Object oriented

hardware in termini di memoria ed efficienza, senza sprechi inutili o, al contrario, rischi di sottostima dei requisiti di sistema.

Grazie alla documentazione del linguaggio UML, infine, diviene ancora più facile effettuare eventuali modifiche future al codice e la comunicazione e l'interazione tra tutte le risorse umane che prendono parte allo sviluppo del sistema è molto più efficiente e diretta.

Di seguito verranno illustrati i diagrammi UML creati. Per la loro modellazione è stato utilizzato il tool UMLet 11.5[9].

UMLet è un programma open-source, scritto nel linguaggio Java, utilizzato per il disegno rapido di diagrammi UML. Dispone di un'interfaccia utente semplice e facile da utilizzare. Offre supporto di disegno per tutti i tipi di diagrammi e permette, inoltre, di poter esportare i diagrammi creati in tanti formati.

2.4.2 Diagramma di Attività

Descrizione

I diagrammi di attività spesso vengono chiamati anche “diagrammi di flusso”, dal momento che consentono di modellare un processo come un'attività costituita da un insieme di nodi connessi da archi. Un diagramma di attività serve per modellare il flusso di lavoro e fornire uno stato di partenza (stato iniziale) ed uno di arrivo (stato finale). Questo tipo di diagramma fornisce una modellazione per la sequenza di operazioni ‘elementari’ che definiscono un'attività più complessa. I componenti principali di tale diagramma UML sono: le attività, i stati e le transizioni. Un'attività rappresenta una determinata azione che deve essere svolta dentro la funzione. Gli stati rappresentano dei valori interni e le transizioni indicano i percorsi del flusso. L'essenza di un buon diagramma di attività è quella di esprimere uno specifico aspetto del comportamento dinamico di un sistema. A tal fine, è necessario che il livello di astrazione contenga la minor quantità di informazioni in modo tale da aumentare la comprensibilità del diagramma. La caratteristica tipica di questi diagrammi è quella di permettere la modellazione di un processo senza dover specificare la struttura statica delle classi e degli oggetti del processo stesso.

Diagramma di Attività

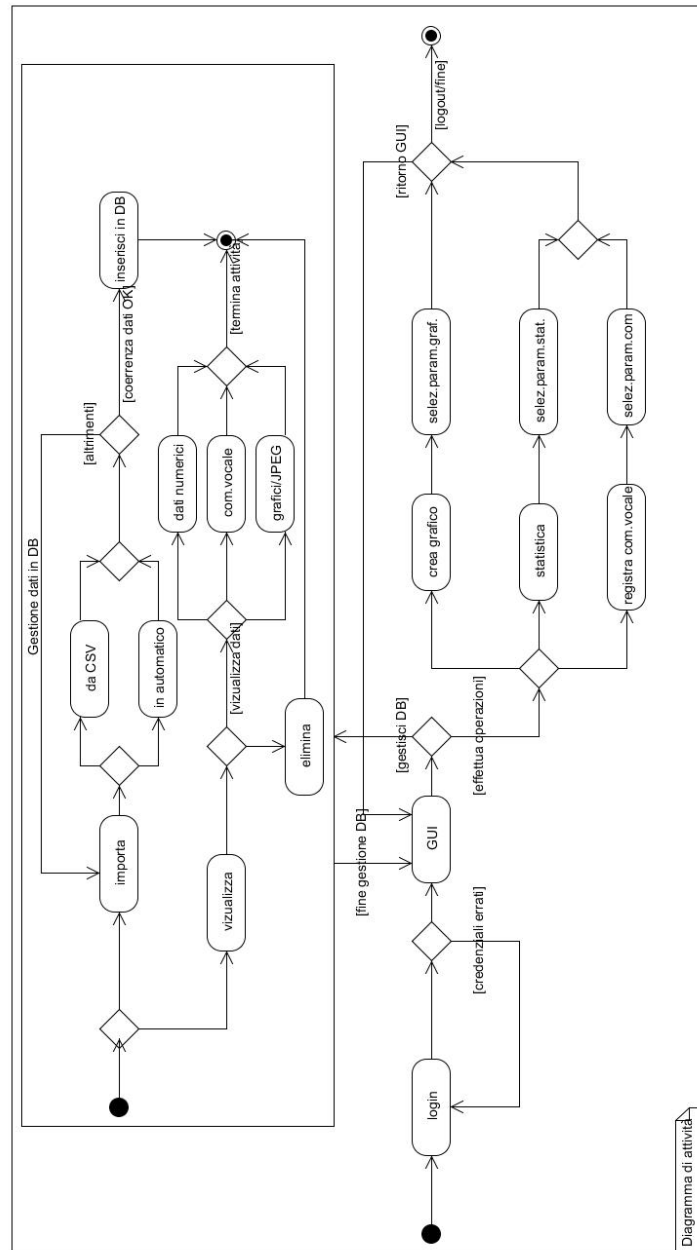


Figura 2.3: Diagramma di Attività

Le attività iniziano con un singolo nodo chiamato nodo iniziale che indica il punto in cui l'esecuzione dell'attività inizia. In particolare, l'attività iniziale è quella in cui l'utente effettua il login tramite l'applicazione. Il flusso passa poi alla verifica delle credenziali da parte del sistema. In caso di errore il flusso ritorna allo stato iniziale. In caso di successo si avanza verso lo stato GUI, che rappresenta l'interfaccia base del sistema. A questo punto, l'utente può scegliere di procedere tra diverse attività. Più dettagliatamente può scegliere di: passare alla gestione dati in database oppure effettuare operazioni di creazione grafici, statistiche o registrazione di commenti vocali. Se si sceglie di creare un grafico si passa all'attività di selezione di parametri grafici, per poi tornare o allo stato GUI o proseguire con l'attività logout. Se si sceglie di effettuare operazioni di statistica si prosegue con l'attività di selezione parametri statistici poi tornare o allo stato GUI o proseguire con l'attività logout. Infine se si decide di registrare un commento vocale si va avanti con la scelta dei parametri relativi a questa attività e si ritorna anche in questo caso allo stato GUI o all'attività di logout. In tutti i casi precedenti se l'attività che segue è quella di logout si giunge al nodo finale in cui termina la sessione utente. Per quanto riguarda invece la gestione del database (diagramma mostrato nella parte superiore), la sessione inizia sempre con un nodo iniziale, ma segue con la selezione delle attività di importazione o visualizzazione dei dati semplici ed oggetti complessi. L'attività di importazione dati può avvenire attraverso le attività in automatico oppure da CSV. A seguire vi è il nodo che gestisce il controllo della coerenza dei dati da inserire nel database. In caso di incoerenza dati, lo stato in cui passa il flusso è di nuovo quello di importazione dei dati. Se i dati sono coerenti, i dati vengono inseriti effettivamente nel database e l'attività termina. Passiamo ora all'attività di visualizzazione dei dati, che se viene selezionata permette la scelta tra le operazioni di cancellazione dei dati visualizzati (nel caso dei dati OHLC) oppure la semplice visualizzazione. Quest'attività può proseguire con la visualizzazione dei risultati statistici salvati, di un grafico oppure con l'ascoltazione di un commento audio. Infine l'attività termina raggiungendo il nodo finale come viene mostrato in figura.

2.4.3 Diagramma dei Casi d'uso

Descrizione

Il diagramma dei casi d'uso permette di visualizzare una tipica interazione tra un utente e il sistema. Questo tipo di diagramma descrive le modalità di utilizzo del sistema da parte dell'utente ma non rivela l'orga-

nizzazione interna del sistema stesso. I diagrammi dei casi d'uso possono essere definiti a livelli diversi. Questo significa che questi tipi di diagrammi possono essere utilizzati non solo per rappresentare l'intero sistema ma anche solo parte di esso. In altre parole tali diagrammi rappresentano l'interfaccia esterna del sistema, e specificano un modulo di requisiti di cosa il sistema deve fare (ma non come!).

Le caratteristiche principali di questi tipi di diagrammi sono:

- Ogni caso d'uso è relativo ad almeno un attore;
- Ogni caso d'uso ha un iniziatore (cioè un attore);
- Ogni caso d'uso porta a un risultato rilevante.

Di fondamentale importanza sono le relazioni all'interno del diagramma. Queste relazioni possono essere di tre tipi:

- «*inclusione*» serve a specificare che l'esecuzione di un caso d'uso implica (sempre) l'esecuzione di un altro caso d'uso;
- «*estensione*» serve a specificare che in certe situazioni (ma non sempre) un caso d'uso sarà esteso da un altro;
- La generalizzazione specifica che un caso d'uso eredita le caratteristiche del «*sovra*» caso d'uso, e può sostituirne alcune o aggiungerne nuove in un modo simile all'ereditarietà tra oggetti.

Diagramma dei Casi d'uso

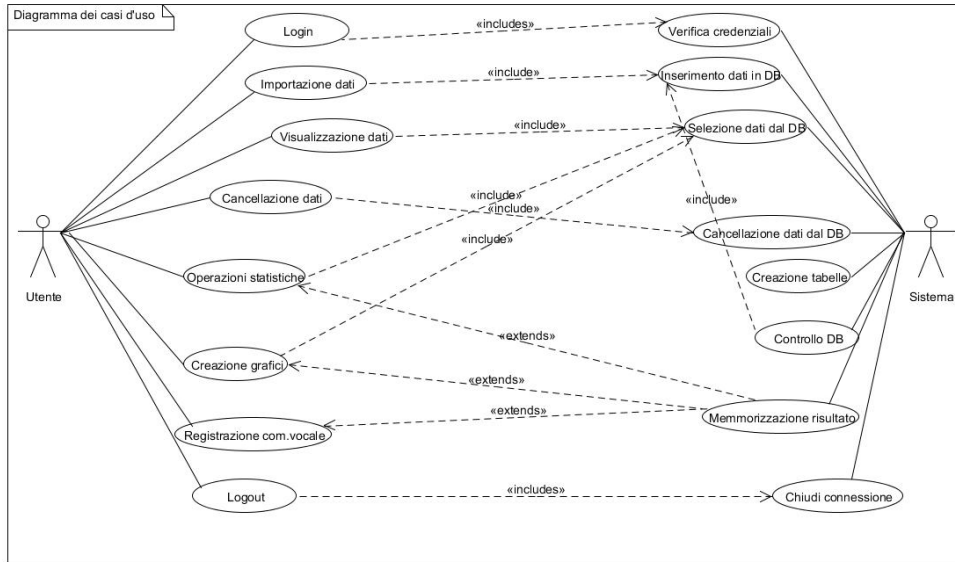


Figura 2.4: Diagramma dei Casi d'uso

Dalla figura si evince che i casi d'uso che l'utente può scegliere di intraprendere sono:

- Login: richiesta di logarsi al database ed iniziare una sessione di lavoro. Questo caso d'uso include a sua volta la verifica dei credenziali di accesso da parte del sistema;
- Importazione dati: richiesta di importare dati OHLC ed inserirli nella base di dati. Il caso d'uso include l'inserimento effettivo dei dati;
- Visualizzazione dati: richiesta dell'utente di visualizzare parte dei dati disponibili. Questo caso d'uso include la selezione, da parte del sistema, di questa porzione di dati;
- Cancellazione dati: richiesta di cancellare dei dati dal database. Questo caso d'uso include la cancellazione effettiva dei dati dalle tabelle della base di dati.
- Operazioni statistiche: richiesta di calcoli statistici da parte dell'utente. Tale caso d'uso include la selezione dei dati su cui si vogliono effettuare le operazioni di statistica ed estende il caso d'uso relativo alla memorizzazione del risultato nella base di dati. Questo perché l'utente può scegliere se salvare i risultati ottenuti oppure scartarli;
- Creazione grafici: richiesta di creazione di un grafico di un determinato tipo. Anche questo caso d'uso, come il precedente, include una selezione di dati ed estende quello responsabile sulla memorizzazione del risultato.

2.4.4 Diagramma di Sequenza

Descrizione

I diagrammi di sequenza illustrano lo scambio di messaggi tra diversi oggetti in una determinata situazione temporale. Gli oggetti sono istanze di classi. I diagrammi di sequenza evidenziano l'ordine in cui i messaggi sono inviati agli oggetti. Nei diagrammi di sequenza gli oggetti sono rappresentati con linee verticali tratteggiate, con il nome dell'oggetto in testa. Anche l'asse temporale si estende temporalmente, in modo tale che i messaggi siano inviati da un oggetto all'altro per mezzo di frecce commentate con l'operazione e il nome del parametro. I messaggi possono o essere sincroni, il tipo usuale di chiamate ai messaggi dove il controllo rimane all'oggetto chiamato fino a quando quel metodo ha finito l'esecuzione, o

asincrono, dove il controllo è ripassato direttamente all'oggetto chiamante. I messaggi sincroni sono rappresentati da un riquadro verticale a lato dell'oggetto chiamato per mostrare il flusso di controllo del programma. Un diagramma di sequenza può essere letto in due modi: in verticale l'asse temporale, mostra l'istante di tempo in cui un oggetto viene chiamato, chiama o muore; in orizzontale gli oggetti e le operazioni che compiono.

Diagramma di Sequenza

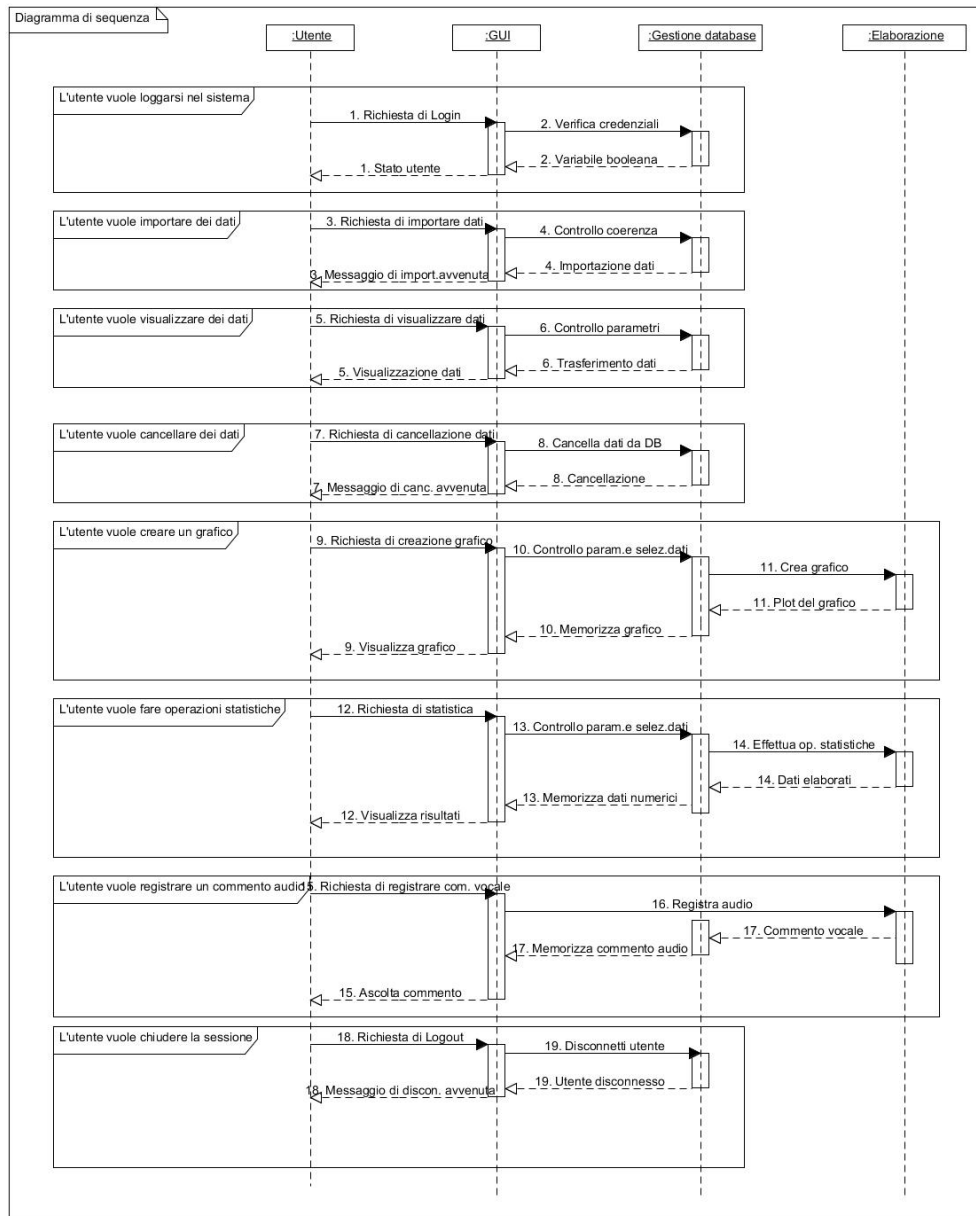


Figura 2.5: Diagramma di Sequenza

La prima fase del diagramma di sequenza mostrato nella figura, è rappresentata dalla richiesta da parte dell'utente di effettuare l'operazione di login. Tale operazione inizia con l'invio di un messaggio di richiesta di login alla componente GUI, la quale a sua volta inoltra la richiesta alla componente responsabile della gestione database, che dopo aver verificato le credenziali di accesso, risponde con un messaggio di successo o di errore. Di conseguenza la componente GUI, in base al tipo di messaggio ricevuto (successo o errore), risponde alla richiesta dell'utente restituendo lo stato corrispondente.

La seconda sequenza di azioni che si evince dalla figura è rappresentata dalla richiesta dell'utente di importare dei dati in database. L'utente invia dunque un messaggio di richiesta di importazione di dati alla GUI, la quale inoltra il messaggio alla componente che gestisce il database, insieme ad una richiesta di controllo di coerenza dei dati. L'ultima componente dopo aver effettuato il controllo, risponde alla GUI, che a sua volta risponde all'utente con un messaggio di successo oppure di errore. In quest'ultimo caso l'utente viene anche informato sulla natura dell'errore.

La terza sequenza, quella che rappresenta la visualizzazione di una porzione di dati inizia con la richiesta dell'utente di poter visualizzare a video dei dati OHLC oppure oggetti complessi. Anche in questo caso la richiesta viene inoltrata alla GUI, per essere poi inoltrata alla componente di gestione database, la quale risponde o con un messaggio di errore oppure con i dati da visualizzare in caso di successo. Per quanto riguarda la sequenza di cancellazione dati il procedimento è molto simile. La differenza principale è che in questo caso la componente che gestisce il database, non risponde inviando dei dati ma solo un messaggio di avvenuta cancellazione o di errore (parametri errati).

Vediamo ora in dettaglio la sequenza che rappresenta la creazione di un grafico. Tale sequenza inizia con la richiesta dell'utente che viene inviata alla componente GUI. Quest'ultima inoltra la richiesta alla componente che gestisce la base di dati che dopo aver recuperati i dati procede con l'inoltrare la richiesta alla componente responsabile alla elaborazione e creazione di oggetti complessi. In caso di successo il flusso passa alla componente di gestione database che effettua eventualmente le operazioni necessarie al salvataggio del grafico creato e inoltra la risposta alla componente GUI che a sua volta restituisce il grafico appena creato all'utente oppure lo informa dell'errore avvenuto.

La sequenza relativa ad una operazione statistica inizia con l'invio di tale messaggio da parte dell'utente alla GUI. La sequenza successiva, cioè lo scambio di messaggi tra la GUI e la componente che gestisce il database è

simile alla sequenza del caso precedente, per quanto riguarda il controllo dei parametri. Cambia ovviamente il messaggio relativo al tipo di operazione da effettuare. Dopo aver ricevuto il messaggio dalla GUI, il componente di elaborazione risponde con il risultato ottenuto in caso di successo oppure con il relativo messaggio di errore, in caso di parametri errati. Anche in questo caso, come in quello relativo ai grafici, i risultati vengono mostrati all'utente da parte della GUI.

La sequenza relativa alla registrazione di un commento audio è diversa dalle precedenti in quanto il flusso passa direttamente dalla GUI (dopo il messaggio dell'utente) alla componente di elaborazione. Questo perché in questo caso la scelta dei parametri non è strettamente necessaria ma permette all'utente di registrare un qualsiasi commento senza fare riferimento ai dati disponibili. La componente di gestione della base di dati viene coinvolta dopo che il commento è stato registrato, nel caso in cui l'utente decida di memorizzarlo.

Infine, la sequenza che descrive l'operazione di logout (fine sessione) inizia sempre con la richiesta dell'utente di dislogarsi dal sistema e finisce con la risposta, da parte della componente che gestisce il database dell'avvenuta disconnessione.

2.4.5 Diagramma di Componenti

Descrizione

Lo scopo di questo tipo di diagramma è la rappresentazione interna del sistema software in termini dei suoi principali componenti. Tali componenti solitamente si cercano tra :

- I file del codice sorgente;
- I componenti run-time;
- Gli eseguibili compilati ed i loro sorgenti.

Tale diagramma è un grafo di componenti collegati tra loro da relazioni di dipendenza. Un componente nello specifico, in questo tipo di diagramma, è una parte fisica e sostituibile di un sistema che contiene l'implementazione, fornendo la realizzazione per un insieme di interfacce. Infatti, un componente può implementare un'interfaccia e può contenere uno o più componenti al suo interno. Il nome di un componente può essere il nome di un file fisico oppure il nome di un sottosistema di un certo che gioca il ruolo di un componente. Per mostrare ulteriori informazioni possono essere

aggiunti dei vincoli ai componenti. Un componente è inteso come un'unità software dotata di una precisa identità e funzionalità.

Diagramma dei componenti

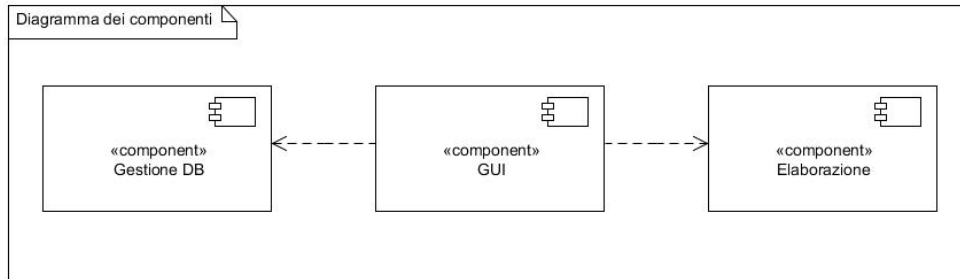


Figura 2.6: Diagramma dei Componenti

Nella realizzazione del progetto, come mostrato nella figura precedente, sono stati creati tre componenti principali, che sono:

- Gestione DB;
- GUI;
- Elaborazione.

Il componente Gestione DB è responsabile appunto della gestione ed organizzazione dei dati all'interno della base di dati. Il secondo componente, GUI, fornisce l'implementazione relativa all'interfaccia grafica del software. Infine, il componente Elaborazione si occupa della gestione delle operazioni sui dati numerici e della creazione di oggetti complessi (grafici e commenti audio).

2.4.6 Diagramma delle Classi

Descrizione

Il diagramma delle classi viene utilizzato per illustrare un'insieme di elementi statici di un modello. Tale diagramma è un grafo che contiene classi e relazioni. Il concetto di "classe" viene usato per descrivere un gruppo di oggetti che sono simili tra di loro in termini di struttura, comportamento e relazioni. Una classe è un descrittore per un'insieme di oggetti che sono tra di loro simili per quanto riguarda la struttura, il comportamento e le

relazioni. Quest'ultime indicano in che modo le classi o gli oggetti sono collegati tra loro. In particolare una relazione può essere:

- Di dipendenza: relazione d'uso, la modifica di un oggetto implica anche la modifica dell'oggetto con il quale si ha tale relazione;
- Di associazione: semplice, aggregazione;
- Di generalizzazione: ereditarietà fra classi;
- Di realizzazione: implementazione interfacce.

Principalmente questo tipo di diagramma viene usato per:

- Individuare i concetti del sistema;
- Specificare tali concetti;
- Specificare le collaborazioni tra gli elementi che compongono il sistema.

Diagramma delle Classi

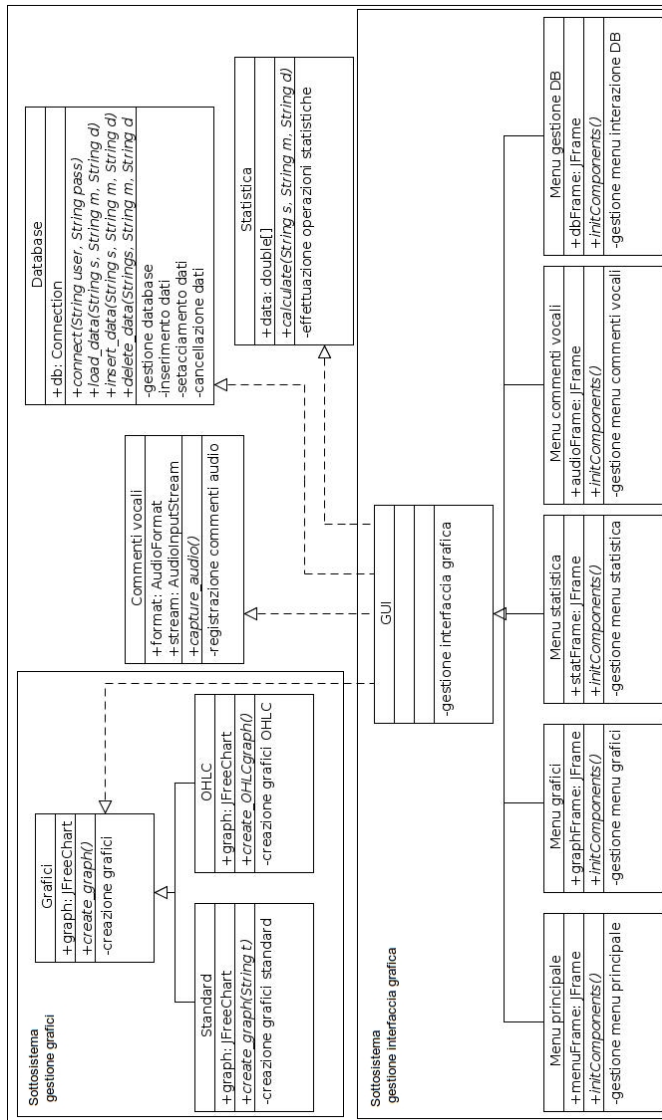


Figura 2.7: Diagramma delle Classi

Dalla figura precedente si evincono a prima vista i due sottosistemi che racchiudono tra loro un'insieme di classi. Questi sono: il sottosistema GUI, responsabile della gestione dell'interfaccia grafica ed il sottosistema che si occupa della gestione grafici. Vediamo ora in dettaglio quali sono le classi che compongono il diagramma e come sono collegate tra loro. Partiamo dalla GUI. Questa è la super classe che fornisce la gestione di base dell'interfaccia dell'applicativo. La classe viene estesa da 5 sotto classi. Tali classi forniscono delle specializzazioni della super classe e ogni una di esse svolge un compito specifico. In particolare, Menu Principale fornisce l'implementazione dell'interfaccia principale del programma, Menu Grafici quella sulla creazione/visualizzazione grafici, Menu Statistica l'interfaccia relativa alle operazioni statistiche, Menu commenti vocali implementa la parte grafica della gestione dei commenti audio, ed infine, Menu gestione DB si occupa di gestire l'interfaccia relativa alla gestione DB.

La classe che gestisce l'interazione con la base di dati viene rappresentata dall'oggetto Database. La variabile pubblica "db" e di tipo "Connection" contiene l'istanza della connessione in uso per una sessione di lavoro. I quattro metodi pubblici: "connect", "load_data", "insert_data" e "delete_data" sono responsabili rispettivamente di: stabilire la connessione, selezionare dei dati restituendoli alla interfaccia grafica, inserire dei dati e cancellare parte di essi. Il primo metodo, ha due parametri che sono lo username e la password che l'utente inserisce per iniziare la sessione. Il secondo ha tre parametri che sono il nome dell'azione, il nome del mercato ed una stringa che contiene informazioni relative all'intervallo di tempo e la frequenza dei dati. Anche gli altri metodi pubblici "insert_data" e "delete_data" hanno ciascuno tre parametri di tipo Stringa, con significato identico a quelli del metodo precedente.

La classe Statistica definisce una variabile pubblica di tipo double[] che contiene l'array dei dati OHLC sui quali l'utente ha scelto di fare operazioni statistiche. Il metodo più importante che la classe implementa è "calculate", che in base ai parametri che gli vengono passati sceglie l'operazione da effettuare e restituisce il risultato oppure i risultati.

La gestione di base relativa ai grafici viene offerta dalla classe Grafici. La variabile pubblica "graph" di tipo JFreeChart che definisce, viene usata per contenere il grafico da creare. Per la creazione del grafico si utilizza il metodo "create_graph". Il tipo di grafico creato da questa classe è quello standard ad una variabile (grafico semplice). Questa classe viene estesa da due sotto classi: Standard e OHLC, responsabili rispettivamente della creazione di grafici standard a più variabili e di grafici OHLC. La variabile pubblica si chiama nello stesso modo su entrambe le classi, "graph" ed è

definita dello stesso tipo. Il metodo “create_graph” della classe Standard fa un override dell’omonimo metodo della super classe Grafici e ne fornisce una specializzazione relativa alla creazione di grafici standard ‘2D ma in più variabili ed anche su più azioni o mercati azionari contemporaneamente. Infine, la classe Commenti vocali gestisce la creazione e la riproduzione di commenti vocali. Definisce due variabili pubbliche: “format” di tipo AudioFormat (il formato del file audio) e “stream” di tipo AudioInputStream (il flusso in ingresso). Il metodo di maggiore importanza si chiama “capture_audio” ed è quello che fornisce l’implementazione per la registrazione di un commento vocale.

2.4.7 Diagramma di Deployment

Descrizione

Il diagramma di deployment specifica l’hardware fisico sui cui verrà eseguito il sistema software ed il modo in cui il software viene collocato su quell’hardware. E’ un diagramma di tipo statico utilizzato per descrivere un sistema in termini di risorse hardware, dette nodi, e di relazioni fra di esse. Il nodo è rappresentato tramite un cubo ed un nome, e raffigura una risorsa hardware disponibile al sistema, mentre le relazioni vengono realizzate attraverso linee che collegano i vari nodi. In generale, il nodo non rappresenta una specifica risorsa, ma una classe di risorse dello stesso tipo, perciò per rappresentare una specifica risorsa del sistema si usano le istanze di nodo.

Oltre ai nodi ed alle relazioni, ulteriori elementi del diagramma sono l’artefatto, che rappresenta una specifica porzione fisica di informazioni utilizzata o prodotta dal processo di sviluppo del software, ed il device, una risorsa fisica computazionale con capacità elaborative sulla quale possono essere allocati artefatti per l’esecuzione.

Diagramma di Deployment

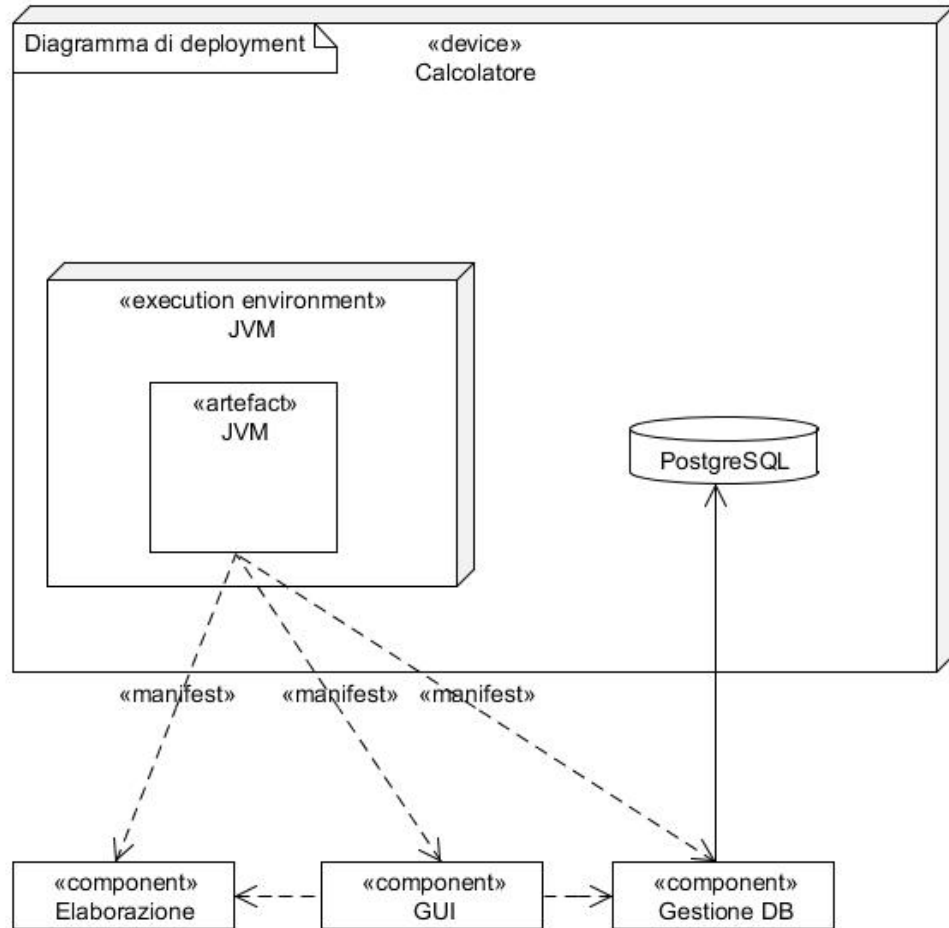


Figura 2.8: Diagramma di Deployment

Nella figura precedente vengono evidenziati i seguenti elementi:

- **Calcolatore:** rappresenta l'hardware fisico su cui viene eseguito l'applicativo;
- **Execution environment:** l'ambiente software necessario al fine di poter permettere una corretta esecuzione del programma. Al suo interno racchiude l'artefatto JVM⁷, che è il componente della piattaforma Java che esegue i programmi scritti in bytecode. Tale artefatto presenta 3 relazioni di dipendenza (manifest) con i componenti GUI, Gestione DB ed Elaborazione. Tali relazioni illustrano gli elementi di modellazione utilizzati nella costruzione dell'artefatto stesso;
- **PostgreSQL:** il DBMS, legato al componente Gestione DB che si occupa del controllo della base di dati.

⁷Java Virtual Machine

Capitolo 3

Fasi di Sviluppo

3.1 Progettazione

La progettazione è stata la fase più importante di tutto il ciclo di sviluppo. In questa fase è stato definito come i requisiti del sistema dovevano essere soddisfatti.

Essendo Java il linguaggio di programmazione, è stato descritto il sistema in termini delle principali classi e delle loro interrelazioni. Si è cercato il più possibile di progettare un sistema con un'architettura semplice ma robusta. Inoltre, è stata dedicata particolare attenzione alle prestazioni ovvero i tempi di esecuzione dell'applicazione. Sono state studiate delle query SQL ottimizzate appositamente per il loro compito all'interno del programma. Le domande principali su cui è stata data risposta prima di procedere con l'implementazione sono state le seguenti:

- Quali responsabilità devono avere le diverse parti del sistema?
- In che modo devono essere collegati queste parti?
- Come fare per rendere il sistema il più stabile possibile?
- Come fare per rendere il sistema il più performante possibile?

3.2 Implementazione

In questa sezione viene descritta l'implementazione vera e propria del sistema, ovvero, la scrittura del codice sorgente. Durante questa fase si è cercato il più possibile di attenersi alle linee guida del paradigma object-oriented per una buona programmazione. Avendo creato in precedenza

i diagrammi UML illustrati nel capitolo precedente e soprattutto avendo fatto una dettagliata progettazione questa fase di sviluppo non è stata eccessivamente complessa.

Prima di iniziare con la scrittura del codice, inoltre, è stato fatto uno schema generale sull'ordine in cui dovevano essere implementate le classi Java che compongono il software. Si è scelto di iniziare con l'implementazione dell'interfaccia grafica. Dopo è stata implementata la classe che gestisce i dati e l'interazione con il database PostgreSQL. Poi, andando per ordine, sono state implementate rispettivamente le classi che gestiscono le operazioni di statistica, creazione/manipolazione grafici, ed infine la classe che gestisce i commenti vocali.

3.2.1 Descrizione delle Classi

Vediamo ora più in dettaglio come è strutturato il sistema. Di seguito vengono elencate le classi principali con una breve descrizione dell'oggetto che rappresentano ed il loro ruolo.

- DBManager: è la classe che gestisce i dati ed il database. Più precisamente si occupa di:
 - Verificare le credenziali di connessione;
 - Stabilire la connessione con il database;
 - Creare le tabelle che compongono il database;
 - Setacciare i dati da inserire;
 - Controllare la coerenza dei dati;
 - Inserire i dati OLHC nelle appropriate tabelle nel database;
 - Garantire in ogni momento l'integrità del database;
 - Restituire i dati richiesti in fase di visualizzazione;
 - Chiudere la connessione al termine della sessione.

- GUI: è la classe che gestisce l'interfaccia grafica del sistema. Questa classe viene ereditata dalle seguenti classi che che la estendono e svolgono dei compiti specializzati
 - MPrin - gestisce l'interfaccia grafica del menu principale del sistema;
 - MGraf - gestisce l'interfaccia grafica per la creazione/visualizzazione grafici;

- MStat - gestisce l'interfaccia grafica per le operazioni statistiche;
 - MComm - gestisce l'interfaccia grafica per la registrazione/ascoltazione dei commenti vocali;
 - MDati - gestisce l'interfaccia grafica per l'interazione dell'utente con il database.
- JPlot: è la classe che gestisce la creazione dei grafici. Questa classe viene ereditata dalle seguenti classi che la estendono e gestiscono specifici tipi di grafici
 - JStand - gestisce i grafici standard;
 - JOhlc - gestisce i grafici OHLC.
 - Statistic: è la classe che gestisce le operazioni di statistica quali:
 - Max;
 - Min;
 - Sum;
 - Variance;
 - Skenwess;
 - Kurtosis;
 - Mean;
 - Sdeviation.

La classe che gestisce la creazione e l'ascolto dei commenti vocali si chiama Comment. Il suo compito è quello di permettere all'utente di registrare un commento vocale, di controllare una serie di parametri quali dimensione del commento e qualità audio e di permettere in un momento successivo la riproduzione del commento.

3.3 Prima Fase di Test

La prima fase di test è stata una parte molto importante dell'intero ciclo di sviluppo. In questa fase sono state testate le funzionalità del sistema per identificare la maggior parte dei bug. Ogni volta che un bug significativo veniva scoperto questo veniva inserito in una lista con una breve descrizione e una possibile localizzazione del problema. Per i bug minori, cioè quelli

immediatamente capiti e localizzati veniva fatta la correzione al momento. Per fare la prima fase di test si è andato per ordine testando le seguenti funzionalità:

1. Corretta visualizzazione dell'interfaccia grafica;
2. La comunicazione del sistema con il database;
3. Creazione delle tabelle primarie;
4. Inserimento corretto dei dati nelle aposite tabelle;
5. Aggiornamento appropriato delle tabelle;
6. Visualizzazione dei dati secondo diversi criteri;
7. Operazioni di statistica;
8. Creazione dei grafici;
9. Memorizzazione degli risultati ottenuti.

3.4 Debugging

In questa fase sono stati corretti tutti i bug rilevanti del sistema che erano stati evidenziati nella fase precedente. Partendo dalla localizzazione del problema, il bug veniva corretto e poi testato con un insieme esaustivo di test creato apposta per il bug in questione. Se la correzione fatta permetteva di passare tutti i test il bug veniva cancellato dall'elenco e si procedeva con la correzione del successivo.

Ovviamente ci sono stati dei casi in cui la localizzazione non era esatta e quindi veniva fatta un'analisi più approfondita ed infine la correzione.

3.5 Seconda Fase di Test

Questa seconda fase di test è stata studiata per rilevare i bug che non sono stati trovati durante la prima fase di test. Per fare questo si è creato un insieme di test diverso dai primi e più approfondito. In particolare si è testato:

1. Controllo coerenza dati;
2. Gestione dei dati duplicati;

3. Creazione/Memorizzazione di commenti vocali;
4. Operazioni di statistica concorrenti;
5. Operazioni di statistica su grandi insiemi di dati;
6. Manipolazione grafici.

Capitolo 4

Realizzazione

4.1 Descrizione dell'applicativo

In questa sezione verrà illustrato l'applicativo creato. Verranno descritte le sue funzionalità ed il suo utilizzo da parte dell'utente finale. Per una maggiore chiarezza verranno inserite delle immagini per presentare le più importanti sezioni del software. In questo modo sarà più chiaro il suo funzionamento e l'interfaccia grafica. L'esecuzione del programma inizia con l'interfaccia di login dove l'utente può inserire il proprio username e password per connettersi con il database ed iniziare ad usare l'applicazione. Di seguito viene mostrato l'interfaccia grafica.

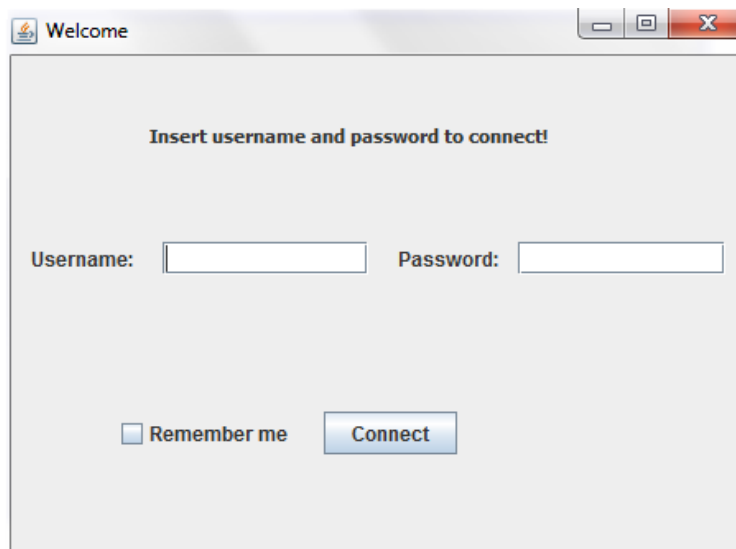


Figura 4.1: Login

L'opzione 'remember me' permette all'utente di poter connettersi senza inserire lo username in futuro. Una volta cliccato il pulsante 'connect', se le credenziali sono corrette l'utente viene avvisato tramite un messaggio di successo. L'avviso è il seguente:

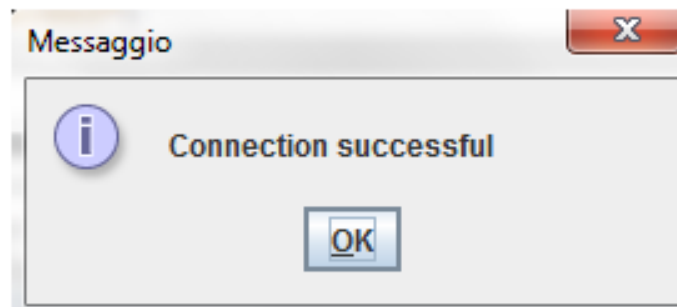


Figura 4.2: Login effettuato

In caso il nome utente oppure la password siano sbagliati, il messaggio di errore è:

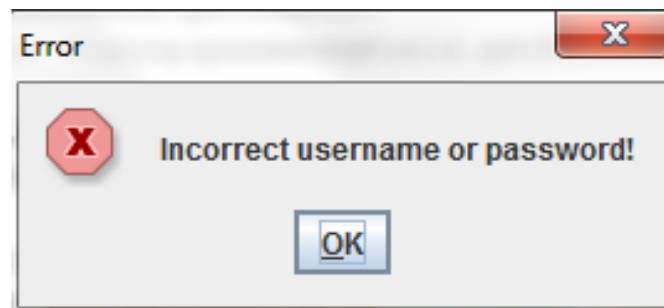


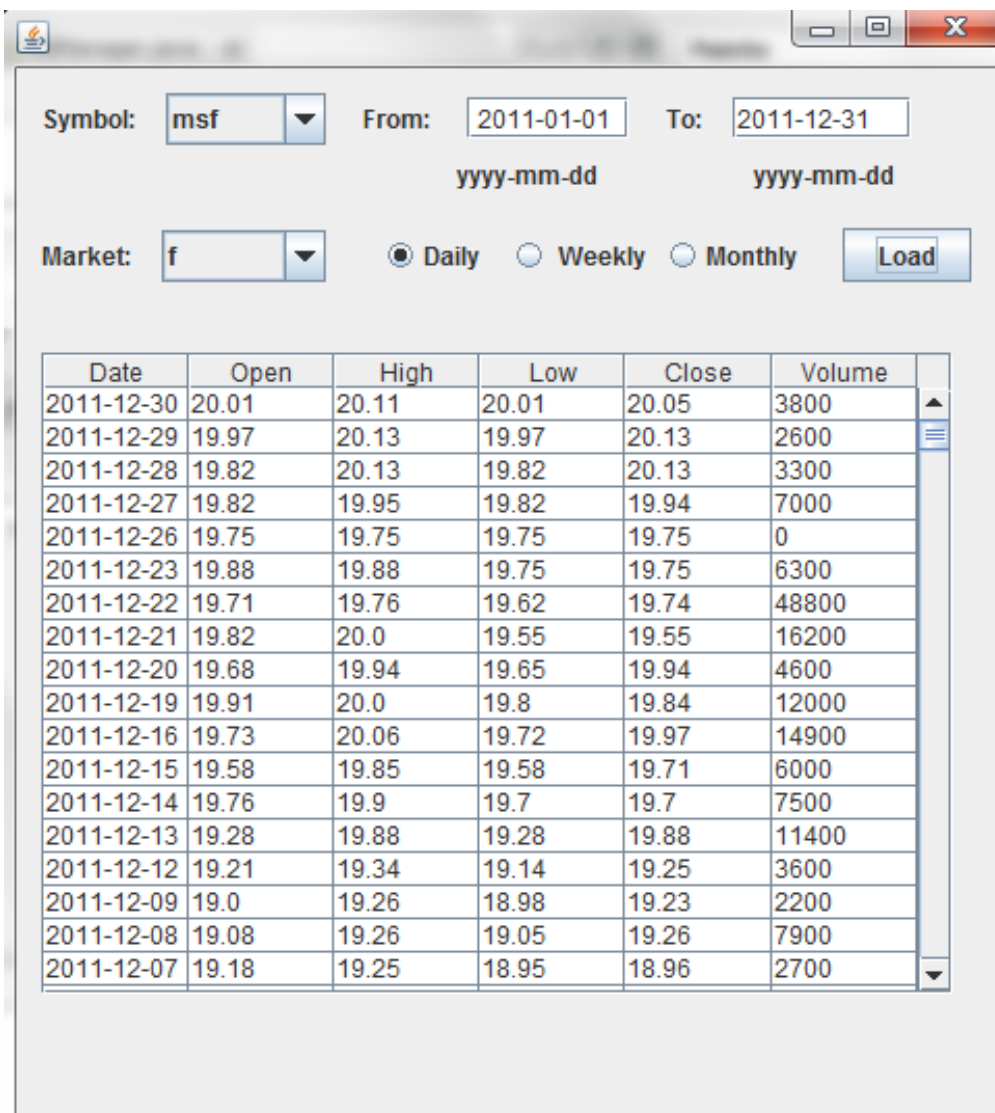
Figura 4.3: Credenziali errate

Dopo che la connessione è stata stabilita viene mostrata l'interfaccia principale del programma. Questo è il punto di partenza. Nella figura di seguito viene mostrato l'interfaccia grafica della schermata principale del programma:



Figura 4.4: Schermata principale

Come si vede nella figura di sopra la bara dei menu principale è composta da tre voci: 'File', 'Tools' e 'About'. Il primo è composto da tre sottomenu: 'Load data...', 'Import', 'Import from file' e 'Exit'. Vediamo in dettaglio le funzionalità di ciascuno di loro. Il primo permette all'utente di visualizzare porzione dei dati OHLC in base a criteri di ricerca personalizzati.



Symbol: From: To:
yyyy-mm-dd yyyy-mm-dd

Market: Daily Weekly Monthly

Date	Open	High	Low	Close	Volume
2011-12-30	20.01	20.11	20.01	20.05	3800
2011-12-29	19.97	20.13	19.97	20.13	2600
2011-12-28	19.82	20.13	19.82	20.13	3300
2011-12-27	19.82	19.95	19.82	19.94	7000
2011-12-26	19.75	19.75	19.75	19.75	0
2011-12-23	19.88	19.88	19.75	19.75	6300
2011-12-22	19.71	19.76	19.62	19.74	48800
2011-12-21	19.82	20.0	19.55	19.55	16200
2011-12-20	19.68	19.94	19.65	19.94	4600
2011-12-19	19.91	20.0	19.8	19.84	12000
2011-12-16	19.73	20.06	19.72	19.97	14900
2011-12-15	19.58	19.85	19.58	19.71	6000
2011-12-14	19.76	19.9	19.7	19.7	7500
2011-12-13	19.28	19.88	19.28	19.88	11400
2011-12-12	19.21	19.34	19.14	19.25	3600
2011-12-09	19.0	19.26	18.98	19.23	2200
2011-12-08	19.08	19.26	19.05	19.26	7900
2011-12-07	19.18	19.25	18.95	18.96	2700

Figura 4.5: Visualizzazione dati OHLC

Il campo 'Symbol' rappresenta il nome del simbolo ovvero dell'azione. Il campo 'Market' il mercato in cui l'azione è quotata ed i campi 'From' e 'To' rispettivamente la data di inizio e la data di fine in cui si vogliono visualizzare i valori OHLC. Infine l'utente può scegliere la frequenza con la quale i dati verranno mostrati. La scelta consiste, al momento, in 'giornalieri', 'settimanali' e 'mensili'. Nel caso in cui i criteri di ricerca siano errati, p.e. dati mancanti, il sistema genera un messaggio di errore mostrando all'utente il parametro o i parametri sbagliati. Per importare dei dati (prezzi storici) l'utente può scegliere il sottomenu 'Import...' o 'Import from file...'. Il primo permette di scaricare i dati direttamente dal sito www.finance.yahoo.com, mentre il secondo permette di caricare un file in formato CSV. Nel primo caso l'utente inserisce il nome dell'azione che sta cercando, il mercato e l'intervallo di tempo. Nel secondo caso invece soltanto i primi due perché l'intervallo viene deciso in automatico dal sistema leggendo il file caricato.

Vediamo di seguito come avviene la scelta tramite l'interfaccia grafica del programma:

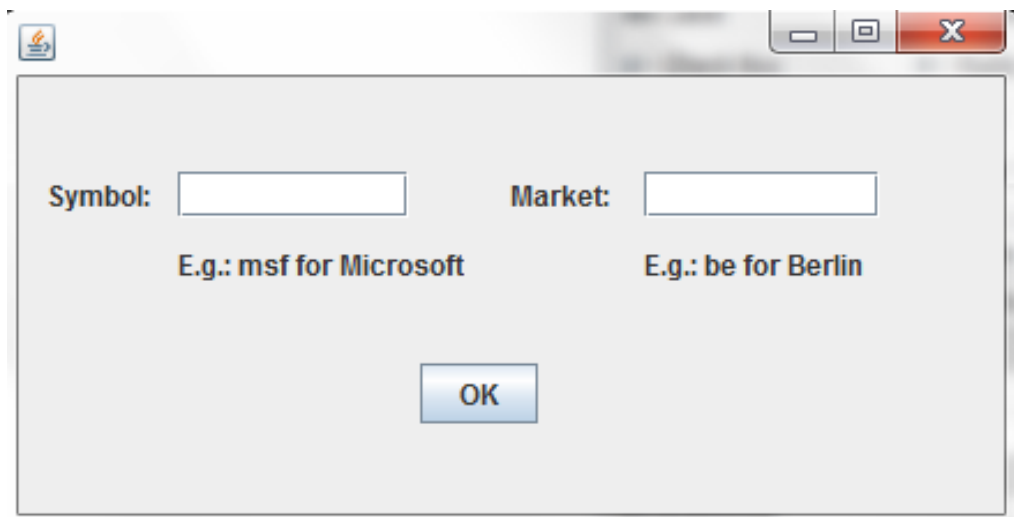


Figura 4.6: Importazione dati da file

La figura mostra la schermata del programma precedente al caricamento del file. L'utente viene invitato a inserire il nome del simbolo con il quale vuole rappresentare l'azione che sta inserendo in database ed il mercato al quale si riferisce. Una volta cliccato sul pulsante 'OK' si apre una finestra dove si può scegliere il file CSV creato in precedenza e caricarlo. A questo punto il sistema analizza il file: determina l'intervallo di tempo dei dati, verifica la correttezza del formato dei valori OHLC, controlla se ci sono dati duplicati ed altri controlli di routine. Se non ci sono errori il sistema inserisce nell'apposita tabella i dati caricati e l'utente viene avvisato da un messaggio di conferma. Nel caso in cui il sistema trovi degli errori, p.e. formato file errato, formato dati errato, etc., viene mostrato un messaggio di errore che avvisa l'utente del problema riscontrato. Per scaricare i dati in automatico la schermata è simile alla precedente. La differenza è che in questo caso l'utente deve scegliere l'intervallo di tempo per il quale vuole scaricare i dati e la frequenza: giornaliera, settimanale o mensile. Anche nella importazione automatica il sistema effettua dei controlli e genera appositi messaggi di errore se l'utente sta cercando di scaricare dati mancanti o non coerenti.

Vediamo ora come avviene la creazione dei grafici. Nei grafici OHLC vengono mostrati i quattro valori OHLC su una linea verticale in un determinato intervallo di tempo scelto dall'utente. L'asse verticale contiene il prezzo del bene trattato su un mercato mentre quello orizzontale il tempo di misura. Per creare un grafico OHLC l'utente sceglie il nome del bene, il mercato e l'unità di tempo su quale vuole visualizzare i valori. Se i parametri sono corretti e i dati disponibili nel database il programma restituisce il grafico OHLC. Per fare dei confronti su più azioni o mercati è possibile creare dei grafici multipli sulla stessa schermata. Ogni uno di questi viene mostrato con un colore diverso e in fondo viene fatta vedere la legenda. Vediamo nella figura di seguito un esempio di un grafico OHLC.

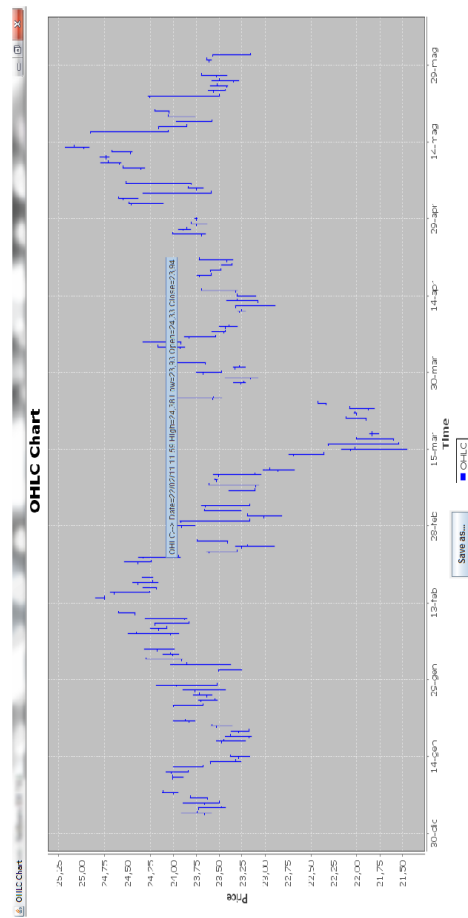
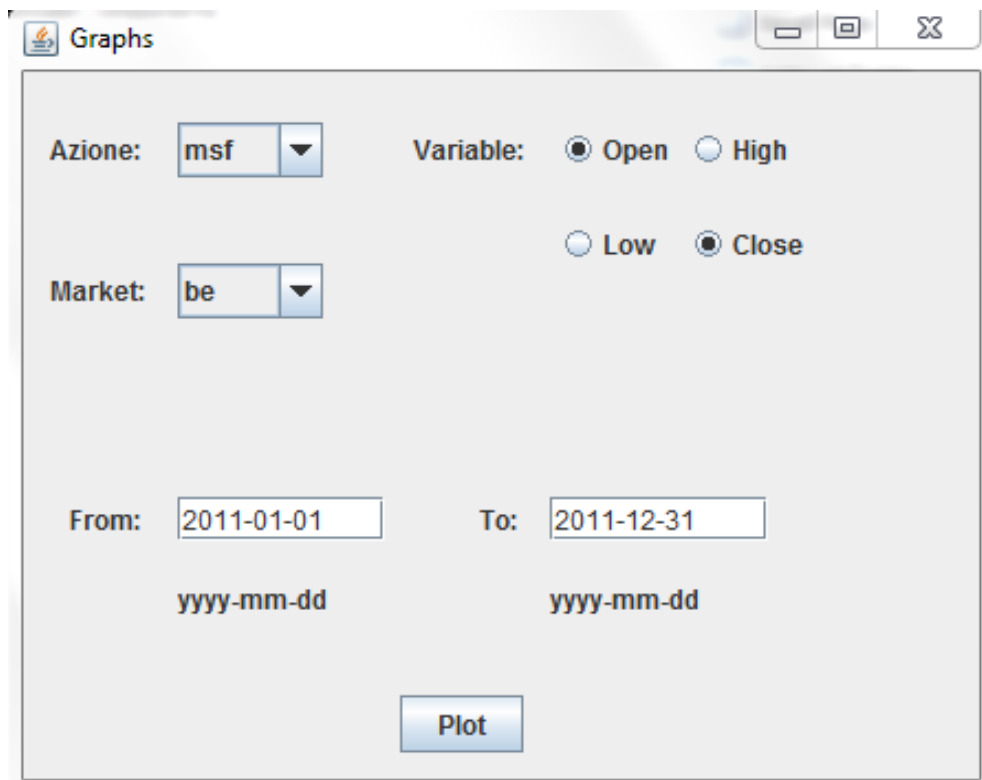


Figura 4.7: Esempio di grafico OHLC

Le quattro linee orizzontali che compaiono su quelle verticali rappresentano i quattro valori OHLC. Se si seleziona con il mouse una linea vengono mostrati le informazioni sulla data e sui valori di tutte le variabili. Cliccando con il tasto destro l'utente può ingrandire a piacere l'immagine. Inoltre c'è la possibilità di salvare il grafico cliccando il pulsante 'Save as...'.
L'utente può dare un nome personalizzato prima di salvare il file, rendendo quindi possibile la facile localizzazione in futuro. Vediamo ora un grafico standard su più variabili. Questo tipo di grafico permette all'utente di scegliere quali su quali valori vuole creare il grafico tra i quattro. La schermata del programma è la seguente:



The screenshot shows a window titled "Graphs" with the following parameters:

- Azione:** msf (dropdown menu)
- Market:** be (dropdown menu)
- Variable:** Open (selected), High, Low, Close (radio buttons)
- From:** 2011-01-01 (text box), with the format yyyy-mm-dd indicated below.
- To:** 2011-12-31 (text box), with the format yyyy-mm-dd indicated below.
- Plot** (button)

Figura 4.8: Parametri grafico standard

Nella figura si vede che è stato scelto di creare un grafico per due variabili, la variabile Open e Close sull'intero anno 2011. Il grafico risultante è il seguente:

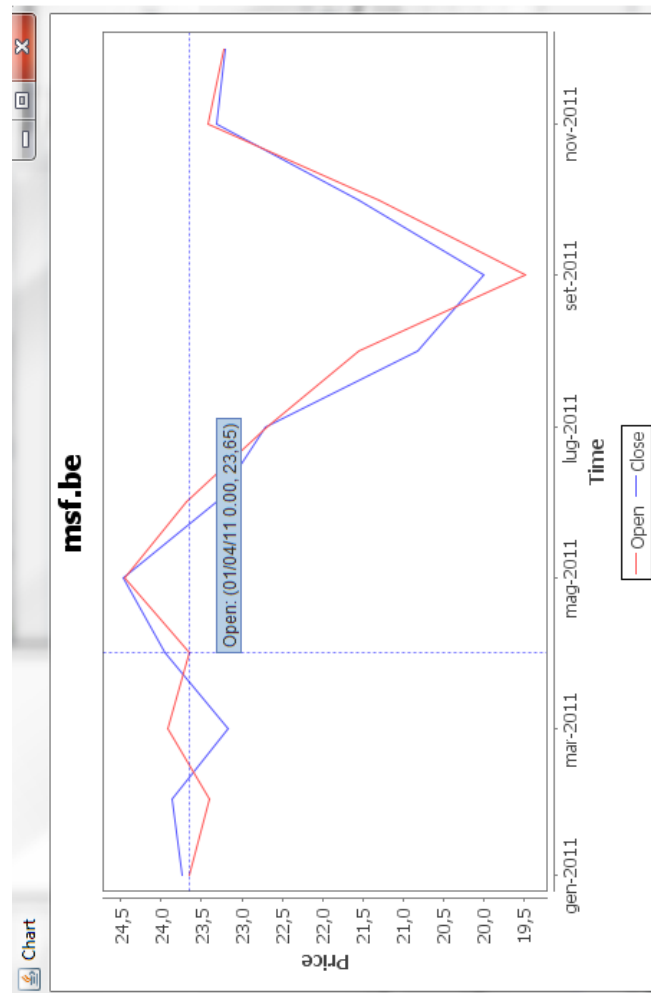
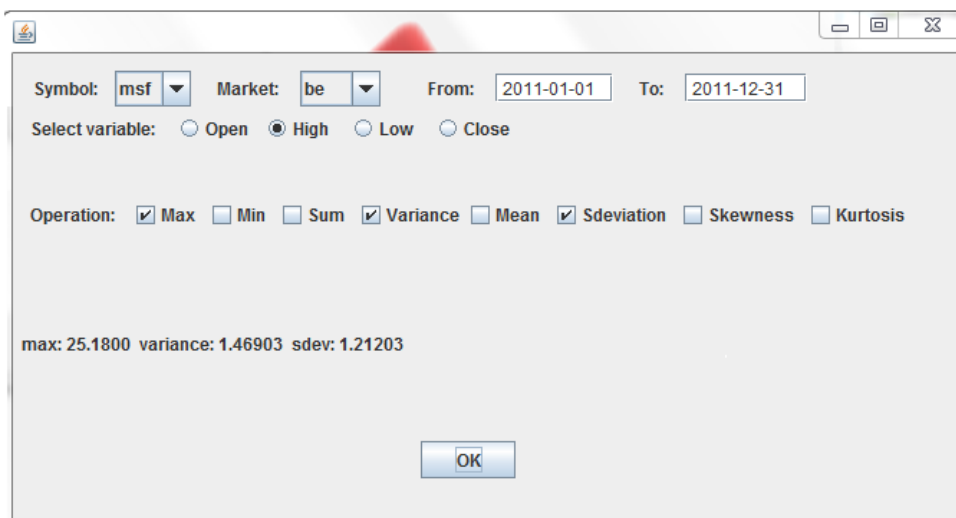


Figura 4.9: Grafico su due variabili

La linea rossa indica l'andamento del valore della variabile Open mentre quella blu della variabile Close. Anche in questo caso cliccando con il tasto destro l'utente ha la possibilità di scegliere tra diverse opzioni di visualizzazione. L'unità di misura del prezzo è la stessa con la quale i dati sono stati importati e inseriti in database. Per visualizzare un grafico salvato in precedenza l'utente può cercare per nome del grafico oppure tramite i dati relativi all'azione, il mercato o l'intervallo di tempo. Passiamo adesso alla sezione che riguarda le operazioni di statistica del software. Di seguito viene mostrata la schermata principale del programma che gestisce le operazioni di statistica singole, ovvero su un'unico titolo e mercato.



Symbol: msf Market: be From: 2011-01-01 To: 2011-12-31

Select variable: Open High Low Close

Operation: Max Min Sum Variance Mean Sdeviation Skewness Kurtosis

max: 25.1800 variance: 1.46903 sdev: 1.21203

OK

Figura 4.10: Statistiche sui dati

L'utente, una volta scelto l'azione ed il mercato insieme all'intervallo di tempo, può selezionare le operazioni statistiche da effettuare. Se si dispongono i dati sull'intervallo richiesto il programma calcola i valori e stampa i risultati. Per fare confronti su più beni o mercati o anche su intervalli di tempo differenti è possibile effettuare operazioni di statistica multipli sulla stessa schermata. Anche in questo caso, come nei grafici i risultati diversi vengono mostrati con colori diversi. Infine vediamo l'interfaccia grafica relativa alla registrazione di commenti vocali.

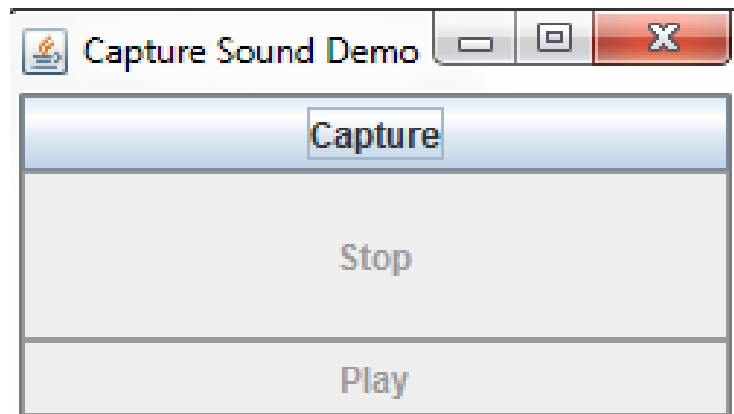


Figura 4.11: Registrazione commenti vocali

L'interfaccia è molto semplice. Essa è composta da un menu con tre pulsanti: 'capture' che inizia la registrazione dell'audio, 'stop' che ferma la registrazione e 'play' per ascoltare il messaggio registrato. Se l'utente decide di chiudere la finestra il programma chiede se vuole che il messaggio venga salvato sul disco. In fase di ascoltazione di un commento l'interfaccia è la medesima ma il pulsante "capture" non è più attivo. In questo caso l'utente può scegliere un nome per il file audio che è stato appena registrato. I dati relativi al commento vengono inoltre salvati nell'apposita tabella in database.

4.2 Misure di prestazione

In questa sezione verranno descritte le prestazioni del sistema. In particolare sono state effettuate misure sui tempi d'esecuzione del programma. Lo strumento utilizzato è lo strumento Profiler dell'ambiente integrato NetBeans. Per quanto riguarda i tempi d'esecuzione i test sono stati fatti su punti 'critici' del programma ovvero la connessione con il database, caricamento dei dati, visualizzazione dei dati e tempo di creazione grafici. Di seguito viene mostrato l'output del profiler.

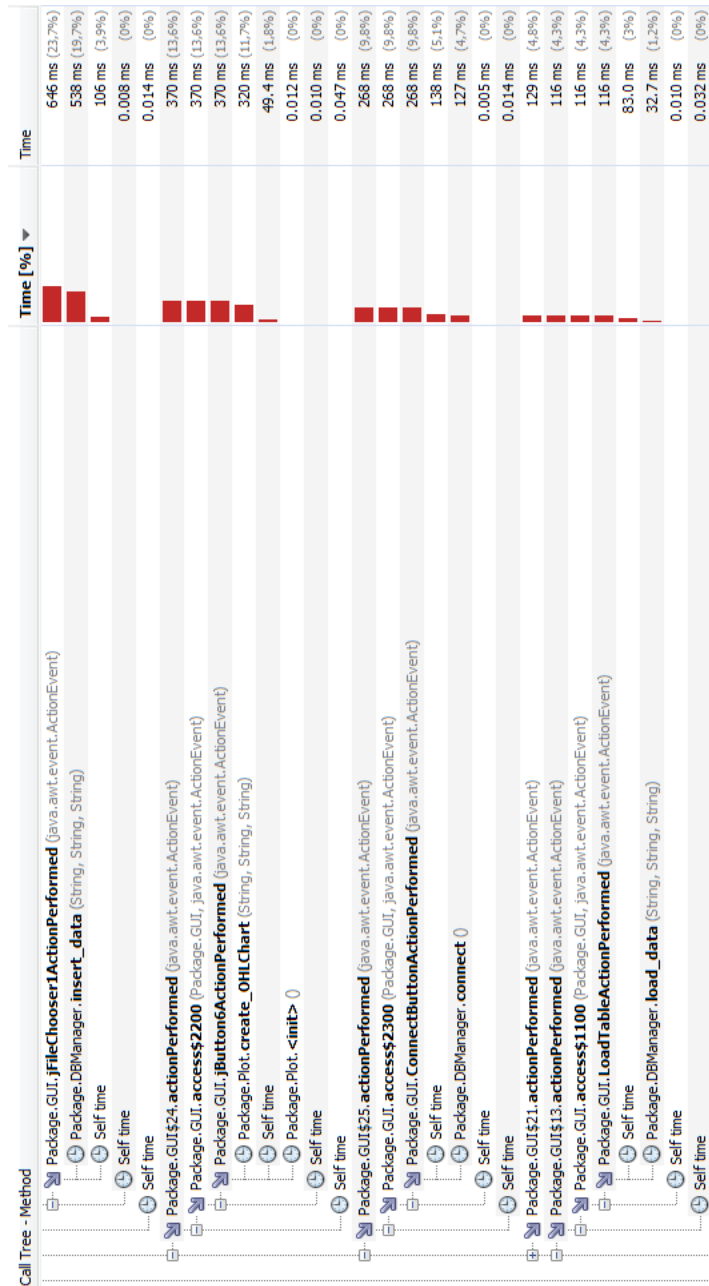


Figura 4.12: Prestazioni del sistema

La prima colonna contiene l'albero delle chiamate dei metodi. La seconda il tempo d'esecuzione in percentuale e l'ultima colonna il tempo in millisecondi.

Dalla figura si evince che il metodo 'connect' responsabile per la connessione con il database impiega **127ms** per verificare le credenziali e stabilire la connessione. L'inserimento dei dati, ovvero il metodo 'insert data' impiega **538ms** per controllare la coerenza ed inserire i dati in database. Nel test in questione sono stati inseriti dati OHLC giornalieri che coprono un arco temporale di 25 anni. La dimensione di tali dati è pari a circa 340K. Per creare un grafico OHLC sul medesimo intervallo di tempo e con la stessa frequenza il sistema impiega **320ms** e per stampare a video questo set di dati il tempo d'esecuzione è pari a **32.7ms**. Considerando la grande mole dei dati che il programma ha gestito durante i test questi tempi sono abbastanza soddisfacenti. Il particolare questo è dovuto all'attenta progettazione del codice ma soprattutto alle query SQL ottimizzate al massimo per fornire il risultato richiesto in minor tempo possibile.

Conclusioni

Scopo di questo lavoro era quello di progettare ed implementare un tool informatico per la gestione dei dati utilizzati nel trading online. Questi dati vengono forniti gratuitamente dal sito <http://www.finance.yahoo.com>. Gli obiettivi da soddisfare comprendevano una gestione efficiente, facile e sicura. Il sistema creato ha un'interfaccia user-friendly per facilitare l'utilizzo da parte degli utenti finali. Sono stati utilizzati soltanto strumenti gratuiti ed open source come il linguaggio di programmazione Java, l'ambiente integrato NetBeans, il database PostgreSQL, Workbench, UMLet. L'architettura del sistema è semplice e robusta. Tutti gli obiettivi preposti all'inizio sono stati raggiunti con successo. Per quanto riguarda gli sviluppi futuri, un'idea da realizzare potrebbe essere quella di adattare il sistema in modo che possa gestire e lavorare con dei dati forniti da banche o istituti di credito che si occupano del trading. Si tratta di dati relativi al prezzo dell'azione non solo su una singola giornata di negoziazione, ma anche ad intervalli di tempo minori, p.es. ogni ora. L'informazione che si può ottenere in questo caso è sicuramente maggiore e può venire incontro a più esigenze che gli investitori possono avere.

Bibliografia

- [1] http://it.wikipedia.org/wiki/Trading_online
- [2] <http://www.borsaitaliana.it/trading-online-expo-2012/homepage/home.htm>
- [3] <http://www.trading-on-line.org/>
- [4] <http://www.tradingonline.pro/>
- [5] <http://www.postgresql.org>
- [6] <http://www.mysql.it/products/workbench/>
- [7] <http://research.microsoft.com/apps/pubs/default.aspx?id=64525>
- [8] <http://oalgo.org/>
- [9] <http://www.umlet.com/>

Elenco delle figure

1.1	Struttura dai	15
2.1	Struttura del database	19
2.2	Gerarchia cartelle	22
2.3	Diagramma di Attività	28
2.4	Diagramma dei Casi d'uso	31
2.5	Diagramma di Sequenza	34
2.6	Diagramma dei Componenti	37
2.7	Diagramma delle Classi	39
2.8	Diagramma di Deployment	42
4.1	Login	51
4.2	Login effettuato	52
4.3	Credenziali errate	52
4.4	Schermata principale	53
4.5	Visualizzazione dati OHLC	55
4.6	Importazione dati da file	56
4.7	Esempio di grafico OHLC	58
4.8	Parametri grafico standard	59
4.9	Grafico su due variabili	60
4.10	Statistiche sui dati	61
4.11	Registrazione commenti vocali	62
4.12	Prestazioni del sistema	64