



LC Elettronica Manager

08/01/2020

Buffa Salvatore
mat. 0697879

Domingo Emanuele
mat. 0696006

In collaborazione con  **LC**elettronica

Abstract

Come progetto valido per l'esame della materia "Programmazione Avanzata" tenuta dal Prof. Gabriele Fici, si è implementato un software gestionale per una Piccola Media Impresa di riparazioni elettroniche. L'applicativo verrà utilizzato dall'attività LC Elettronica di Canale Luca, ed è stato progettato secondo i suoi bisogni. Dopo un'analisi dei requisiti, si è steso il modello UML del progetto utilizzando diversi design patterns per poi concludere con l'implementazione del codice.

LC Elettronica Manager

Clienti Riparazioni Prodotti Attività

Ricerca: ☒ Persone ☐ Aziende

nome	cognome	Codice Fiscale	PIVA	Codice Identificativo	mail	PEC	Telefono	indirizzo	indirizzo spedizione	operation
aaaaa										Dettagli

Annulla Inserisci Esporta dati Importa dati

Indice

1 Introduzione

2 Storia Utente

- 2.1 Gestione clienti
- 2.2 Gestione prodotti
- 2.3 Gestione riparazioni
- 2.4 Gestione attività

3 Analisi dei requisiti

- 3.1 Deduzione dei requisiti
- 3.2 Requisiti funzionali
- 3.3 Requisiti non funzionali

4 Descrizione software

5 UML

6 Design Patterns Utilizzati

- 6.1 Factory
- 6.2 Builder
- 6.3 Observer
- 6.4 Singleton
- 6.5 Memento

7 Conclusioni

- 7.1 Commenti sul risultato finale
- 7.2 Possibili sviluppi futuri
- 7.3 Referenze

1. Introduzione

La realizzazione di questo progetto è stata suddivisa in tre diverse fasi: la prima riguarda la creazione di storie utente, la seconda di generazione dei requisiti a partire da una descrizione da parte del cliente, a cui abbiamo aggiunto le informazioni scaturite dalle nostre domande. L'ultima fase consiste nella scrittura del codice.

Il software così come verrà descritto successivamente dispone di una grafica molto semplice, incentrata sulle funzionalità da offrire al cliente. Nello specifico la prima finestra che viene mostrata al cliente al suo primo avvio è una finestra di info, in cui si specifica il percorso della cartella dove risiederanno tutte le informazioni dell'applicativo, appositamente creata al primo avvio.

Subito dopo si aprirà il programma che molto intuitivamente permetterà al cliente di svolgere tutte le mansioni richieste durante la fase di deduzione dei requisiti. E' presente una toolbar in alto con 4 bottoni, uno per sezione per poter passare da un pannello all'altro. Ogni pannello contiene una barra di filtraggio, una tabella con le informazioni memorizzate e dei bottoni per poter effettuare le operazioni di inserimento, modifica, cancellazione.

Oltre alle operazioni CUD (Create, Uppdate, Delete) in ogni pannello ci sarà un tasto 'Annulla' per annullare l'ultima operazione, e due tasti 'Esporta' ed 'Importa' per esportare/importare il foglio di calcolo generato.

Additionalmente nel pannello delle attività è presente il tasto 'Mostra grafico' per visualizzare una finestra con il report entrate/uscite di una certa tipologia di attività.

2. Storia Utente

2.1 Gestione clienti

Schermata principale

Luca è il proprietario dell'azienda e desidera gestire le operazioni di inserimento, modifica e cancellazione di un cliente. A schermo viene visualizzata la schermata principale e quando fa clic sul campo "Clienti" viene visualizzata la lista dei clienti attualmente presenti all'interno del sistema ed in basso quattro pulsanti "Annulla", "Inserisci", "Importa dati", "Esporta dati".

Se sceglie "Annulla" viene annullata l'ultima operazione effettuata nel sistema.

Se sceglie "Inserisci" il sistema mostra una nuova finestra in sovrapposizione alla finestra principale, che permette a Luca di inserire un nuovo cliente compilando tutti i dati presenti nella finestra (Nome, Cognome, etc.).

Se sceglie "Importa dati" il sistema mostra una finestra in cui Luca può scegliere graficamente, dunque scorrendo tra le varie cartelle, il file da utilizzare per importare i dati dei clienti.

Se sceglie "Esporta dati" il sistema mostra una finestra in cui Luca può scegliere graficamente, dove salvare il file contenente tutte le informazioni dei clienti memorizzati all'interno del sistema.

Modifica

Luca desidera modificare un cliente già presente nel sistema. Davanti a lui è presente la schermata principale in cui facendo clic sul pulsante posto in alto "Clienti" viene mostrata la schermata con tutti i clienti memorizzati nel sistema. Luca ha la possibilità di scorrere lungo la lista dei clienti o cercarne uno in particolare mediante la barra di ricerca posta in alto. Una volta trovato il cliente da modificare, dovrà spostarsi lungo il margine destro in corrispondenza del cliente e fare clic sul pulsante "Dettagli". Una volta fatto ciò, verrà mostrata una finestra in sovrapposizione della principale, contenenti tutti i dati del cliente ed una checkbox in basso con la dicitura "Modifica" che permette a Luca di modificare i dati del cliente. Una volta modificati tutti i dati, mediante il pulsante "Applica" il sistema sovrascrive le informazioni del cliente e la finestra dei dettagli viene chiusa.

Cancellazione

Luca desidera cancellare un cliente già presente nel sistema. Davanti a lui è presente la schermata principale in cui facendo clic sul pulsante posto in alto "Clienti" viene mostrata la schermata con tutti i clienti memorizzati nel sistema. Luca ha la possibilità di scorrere lungo la lista dei clienti o cercarne uno in particolare mediante la barra di ricerca posta in alto. Una volta trovato il cliente da modificare, dovrà spostarsi lungo il margine destro in corrispondenza del cliente e fare clic sul pulsante "Dettagli". Una volta fatto ciò, verrà mostrata una finestra in sovrapposizione della principale, contenente tutti i dati del

cliente in questione. Per cancellare tale cliente, dovrà premere il pulsante in basso con la dicitura "Cancella", fatto ciò la schermata temporanea viene chiusa e il cliente cancellato.

2.2 Gestione prodotti

Schermata principale

Luca è il proprietario dell'azienda e desidera gestire le operazioni di inserimento, modifica e cancellazione di un prodotto. A schermo viene visualizzata la schermata principale e quando fa clic sul campo "Prodotti" viene visualizzata la lista dei prodotti attualmente presenti all'interno del sistema ed in basso quattro pulsanti "Annulla", "Inserisci", "Importa dati", "Esporta dati".

Se sceglie "Annulla" viene annullata l'ultima operazione effettuata nel sistema.

Se sceglie "Inserisci" il sistema mostra una nuova finestra in sovrapposizione alla finestra principale, che permette a Luca di inserire un nuovo prodotto compilando tutti i dati presenti nella finestra (Numero seriale, titolo, descrizione, etc).

Se sceglie "Importa dati" il sistema mostra una finestra in cui Luca può scegliere graficamente, dunque scorrendo tra le varie cartelle, il file da utilizzare per importare i dati dei prodotti.

Se sceglie "Esporta dati" il sistema mostra una finestra in cui Luca può scegliere graficamente, dove salvare il file contenente tutte le informazioni dei prodotti memorizzati all'interno del sistema.

Modifica

Luca desidera modificare un prodotto già presente nel sistema. Davanti a lui è presente la schermata principale in cui facendo clic sul pulsante posto in alto "Prodotti" viene mostrata la schermata con tutti i prodotti memorizzati nel sistema. Luca ha la possibilità di scorrere lungo la lista dei prodotti o cercarne uno in particolare mediante la barra di ricerca posta in alto. Una volta trovato il prodotto da modificare, dovrà spostarsi lungo il margine destro in corrispondenza del prodotto e fare clic sul pulsante "Dettagli". Una volta fatto ciò, verrà mostrata una finestra in sovrapposizione della principale, contenenti tutti i dati del prodotto ed una checkbox in basso con la dicitura "Modifica" che permette a Luca di modificare i dati del prodotto. Una volta modificati tutti i dati, mediante il pulsante "Applica" il sistema sovrascrive le informazioni del prodotto e la finestra dei dettagli viene chiusa.

Cancellazione

Luca desidera cancellare un prodotto già presente nel sistema. Davanti a lui è presente la schermata principale in cui facendo clic sul pulsante posto in alto "Prodotti" viene mostrata la schermata con tutti i prodotti memorizzati nel sistema. Luca ha la possibilità

di scorrere lungo la lista dei prodotti o cercarne uno in particolare mediante la barra di ricerca posta in alto. Una volta trovato il prodotto da cancellare, dovrà spostarsi lungo il margine destro in corrispondenza del prodotto e fare clic sul pulsante "Dettagli". Una volta fatto ciò, verrà mostrata una finestra in sovrapposizione della principale, contenente tutti i dati del prodotto in questione. Per cancellare tale prodotto, dovrà premere il pulsante in basso con la dicitura "Cancella", fatto ciò la schermata temporanea viene chiusa e il prodotto cancellato.

2.3 Gestione riparazioni

Schermata principale

Luca è il proprietario dell'azienda e desidera gestire le operazioni di inserimento, modifica e cancellazione di una riparazione, inoltre desidera poter stampare la fattura di una riparazione. A schermo viene visualizzata la schermata principale e quando fa clic sul campo "Riparazioni" viene visualizzata la lista delle riparazioni attualmente presenti all'interno del sistema ed in basso quattro pulsanti "Annulla", "Inserisci", "Importa dati", "Esporta dati".

Se sceglie "Annulla" viene annullata l'ultima operazione effettuata nel sistema.

Se sceglie "Inserisci" il sistema mostra una nuova finestra in sovrapposizione alla finestra principale, che permette a Luca di inserire un nuovo cliente compilando tutti i dati presenti nella finestra (Nome, Cognome, etc.).

Se sceglie "Importa dati" il sistema mostra una finestra in cui Luca può scegliere graficamente, dunque scorrendo tra le varie cartelle, il file da utilizzare per importare i dati dei clienti.

Se sceglie "Esporta dati" il sistema mostra una finestra in cui Luca può scegliere graficamente, dove salvare il file contenente tutte le informazioni dei clienti memorizzati all'interno del sistema.

Modifica

Luca desidera modificare una riparazione già presente nel sistema. Davanti a lui è presente la schermata principale in cui facendo clic sul pulsante posto in alto "Riparazioni" viene mostrata la schermata con tutte le riparazioni memorizzati nel sistema. Luca ha la possibilità di scorrere lungo la lista delle riparazioni o cercarne una in particolare mediante la barra di ricerca posta in alto. Una volta trovata la riparazione da modificare, dovrà spostarsi lungo il margine destro in corrispondenza della riparazione e fare clic sul pulsante "Dettagli". Una volta fatto ciò, verrà mostrata una finestra in sovrapposizione della principale, contenenti tutti i dati del prodotto ed una checkbox in basso con la dicitura "Modifica" che permette a Luca di modificare i dati della riparazione. Una volta modificati tutti i dati, mediante il pulsante "Applica" il sistema sovrascrive le informazioni della riparazione e la finestra dei dettagli viene chiusa.

Cancellazione

Luca desidera cancellare una riparazione già presente nel sistema. Davanti a lui è presente la schermata principale in cui facendo clic sul pulsante posto in alto "Riparazioni" viene mostrata la schermata con tutte le riparazioni memorizzate nel sistema. Luca ha la possibilità di scorrere lungo la lista delle riparazioni o cercarne una in particolare mediante la barra di ricerca posta in alto. Una volta trovata la riparazione da cancellare, dovrà spostarsi lungo il margine destro in corrispondenza della riparazione e fare clic sul pulsante "Dettagli". Una volta fatto ciò, verrà mostrata una finestra in sovrapposizione della principale, contenente tutti i dati della riparazione in questione. Per cancellare tale riparazione, dovrà premere il pulsante in basso con la dicitura "Cancella", fatto ciò la schermata temporanea viene chiusa e la riparazione cancellata.

Generazione PDF

Luca desidera stampare la fattura di una riparazione già presente nel sistema. Davanti a lui è presente la schermata principale in cui facendo clic sul pulsante posto in alto "Riparazioni" viene mostrata la schermata con tutte le riparazioni memorizzate nel sistema. Luca ha la possibilità di scorrere lungo la lista delle riparazioni o cercarne una in particolare mediante la barra di ricerca posta in alto. Una volta trovata la riparazione da cancellare, dovrà spostarsi lungo il margine destro in corrispondenza della riparazione e fare clic sul pulsante "Dettagli". Una volta fatto ciò, verrà mostrata una finestra in sovrapposizione della principale, contenente tutti i dati della riparazione in questione. Per stampare la fattura di tale riparazione, dovrà premere il pulsante in basso con la dicitura "Scarica contratto", fatto ciò la il sistema mostrerà una finestra di dialogo in cui sarà possibile selezionare il percorso dove salvare il PDF. Selezionato tale percorso verrà chiusa la finestra di dialogo e la finestra di dettaglio.

2.4 Gestione attività


Schermata principale

Luca è il proprietario dell'azienda e desidera gestire le operazioni di inserimento, modifica e cancellazione di una attività, inoltre per le attività desidera ricevere un grafico con le entrate ed uscite. A schermo viene visualizzata la schermata principale e quando fa clic sul campo "Attività" viene visualizzata la lista delle attività attualmente presenti all'interno del sistema ed in basso cinque pulsanti "Annulla", "Inserisci", "Importa dati", "Esporta dati", "Mostra grafico".

Se sceglie "Annulla" viene annullata l'ultima operazione effettuata nel sistema.

Se sceglie "Inserisci" il sistema mostra una nuova finestra in sovrapposizione alla finestra principale, che permette a Luca di inserire una nuova attività compilando tutti i dati presenti nella finestra (Descrizione, tipo, data, etc).

Se sceglie "Importa dati" il sistema mostra una finestra in cui Luca può scegliere graficamente, dunque scorrendo tra le varie cartelle, il file da utilizzare per importare i dati delle attività.



Se sceglie “Esporta dati” il sistema mostra una finestra in cui Luca può scegliere graficamente, dove salvare il file contenente tutte le informazioni delle attività memorizzate all’interno del sistema.

Se sceglie “Mostra grafico” il sistema mostra una nuova finestra in cui Luca può visualizzare entrate ed uscite per una data attività selezionando da un apposito menù a tendina tale attività.

Modifica

Luca desidera modificare un'attività già presente nel sistema. Davanti a lui è presente la schermata principale in cui facendo clic sul pulsante posto in alto “Attività” viene mostrata la schermata con tutte le attività memorizzate nel sistema. Luca ha la possibilità di scorrere lungo la lista delle attività o cercarne uno in particolare mediante la barra di ricerca posta in alto. Una volta trovato il prodotto da modificare, dovrà spostarsi lungo il margine destro in corrispondenza della attività e fare clic sul pulsante “Dettagli”. Una volta fatto ciò, verrà mostrata una finestra in sovrapposizione della principale, contenenti tutti i dati della attività ed una checkbox in basso con la dicitura “Modifica” che permette a Luca di modificare i dati della attività. Una volta modificati tutti i dati, mediante il pulsante “Applica” il sistema sovrascrive le informazioni dell'attività e la finestra dei dettagli viene chiusa.

Cancellazione

Luca desidera cancellare un'attività già presente nel sistema. Davanti a lui è presente la schermata principale in cui facendo clic sul pulsante posto in alto “Attività” viene mostrata la schermata con tutte le attività memorizzate nel sistema. Luca ha la possibilità di scorrere lungo la lista delle attività o cercarne una in particolare mediante la barra di ricerca posta in alto. Una volta trovata l'attività da cancellare, dovrà spostarsi lungo il margine destro in corrispondenza dell'attività e fare clic sul pulsante “Dettagli”. Una volta fatto ciò, verrà mostrata una finestra in sovrapposizione della principale, contenente tutti i dati dell'attività in questione. Per cancellare tale attività, dovrà premere il pulsante in basso con la dicitura “Cancella”, fatto ciò la schermata temporanea viene chiusa e l'attività cancellata.

3. Analisi dei requisiti

In questa fase avviene la raccolta dei requisiti tramite colloquio diretto e/o indiretto con il cliente. La prima raccolta è avvenuta mediante colloquio diretto con il cliente, ovvero, siamo andati fisicamente dal cliente ed abbiamo registrato tutte quelle che sono state le richieste da parte di quest'ultimo, effettuando anche noi alcune domande per agevolarci nelle fasi successive. I colloqui successivi, causa festività e/o impegni da parte del cliente, sono stati fatti mediante meeting o chat private, così da poter mostrare al cliente passo passo i progressi fatti e constatare se tutto veniva fatto secondo le sue esigenze. Una volta terminata tutta la raccolta dei requisiti, essi sono stati suddivisi in requisiti funzionali e non funzionali.

3.1 Deduzione dei requisiti

Come accennato precedentemente, la deduzione dei requisiti è avvenuta in mediante colloquio diretto e/o indiretto. Possiamo dunque affermare di aver utilizzato la tecnica dell'intervista rispetto all'etnografia. Quest'ultima non è stata possibile a causa dei vari impegni del cliente e delle festività, che hanno impedito l'immersione da parte nostra nell'ambiente di lavoro in cui il sistema verrà utilizzato. Fortunatamente grazie all'utilizzo dell'intervista aperta e chiusa, è stato possibile acquisire un gran numero di informazioni utili alla raccolta dei requisiti. Di seguito verrà descritto il testo risultante dall'intervista aperta e le domande poste al cliente per l'acquisizione dei requisiti.

Intervista aperta

"Mi servirebbe un gestionale per la mia attività, non mi bastano più i fogli excel. Mi basta un'applicazione semplice e veloce che tenga traccia dei dati anagrafici dei clienti del negozio, devo poterli inserire e rimuovere. Inoltre vorrei la possibilità di modificarli se commetto eventuali errori di battitura. Oltre ad essi, vorrei anche memorizzare tutte le riparazioni che faccio per i clienti memorizzando i dati che vi comunicherò per mail o messaggio, a parte le operazioni di inserimento, modifica e rimozione vorrei anche poter scaricare il contratto che faccio firmare prima di ogni lavoro, per non doverlo compilare io a mano. Analogamente ai clienti, vorrei anche tenere traccia dei vari oggetti che ho in magazzino: dai prodotti per le riparazioni ai piccoli gadget che vendo. Siccome oltre alle riparazioni ho in mente di espandere i servizi offerti dal negozio, vorrei una schermata dove poter inserire le vendite di altri servizi. Sarebbe utile anche una sorta di grafico per vedere se la singola attività ha delle entrate positive o ci sto perdendo. Essendo comunque legato ai fogli di calcolo, vorrei la possibilità di esportare tutti i dati in un formato

compatibile con Excel, meglio se open source perché non escludo l'uso di qualche ambiente Linux, se volessi in futuro stampare questi dati o utilizzare qualche script esterno. Ovviamente vorrei poter filtrare i dati memorizzati e se possibile anche selezionarli col mouse."

Intervista chiusa

Domanda n.1

Team di sviluppo: "Nell'inserimento dei campi dei vari elementi, preferisci qualche controllo sulla sintassi o sull'integrità dei dati?"

Luca: "No assolutamente. Vorrei totale libertà sui dati che inserisco, di alcuni clienti potrei anche voler memorizzare solo il nome, o un nickname o mettere dati fittizi. L'unico campo che gradirei controllare è quello sul numero della riparazione: andrebbe bene un formato 001/2020, con il numero incrementale. Tuttavia vorrei poter eventualmente modificare questo numero manualmente quindi non deve essere completamente automatizzato."

Domanda n.2

Team di sviluppo: "I gli elementi di cui memorizzare i dati (clienti, magazzino, etc) sono tutti uguali? o esistono delle sottocategorie?"

Luca: "Ho due tipo di clienti, persone fisiche o aziende, quindi gradirei questa distinzione. Per il resto, essendo tutto molto dinamico e variabile gradirei soltanto poter inserire io la tipologia dei servizi che offro."

Domanda n.3

Team di sviluppo: "Su che macchina verrà eseguita l'applicazione? Qual è il sistema operativo?"

Luca: "In negozio ho un computer da ufficio, non con prestazioni eccelse, motivo per cui gradirei un sistema semplice. Il sistema operativo è windows 10."

Domanda n.4

Team di sviluppo: "E' presente una connessione ad internet o un server in locale? Potremmo salvare tutti i dati in un database mysql."

Luca: "Non ho nessun serve e la connessione è presente ma non sempre garantita, essendo un servizio satellitare. Quindi se possibile vorrei poter eseguire l'applicazione offline."

Domanda n.5

Team di sviluppo: "Dato che ad ogni riparazione è associato un cliente, se il cliente associato venisse modificato o eliminato, la riparazione deve essere aggiornata o no?"

Luca: "Assolutamente no, una volta che scarico e faccio firmare il contratto i dati non possono cambiare, avrò cura io di conservare tutti i contratti che faccio firmare."

3.2 Requisiti funzionali

- Inserimento, modifica, cancellazione clienti
- Inserimento, modifica, cancellazione aziende
- Inserimento, modifica, cancellazione prodotti
- Inserimento, modifica, cancellazione, visualizzazione grafico attività
- Inserimento, modifica, cancellazione, download pdf del contratto di riparazione
- Per le riparazioni la presenza di un numero univoco del formato XXX/AAAA
- Nel caso di cancellazione di un cliente presente nella riparazione, tale cliente non deve essere eliminato nella riparazione
- Nel caso di modifica di un cliente presente nella riparazione, tale cliente non deve essere modificato nella riparazione
- Per ognuna delle entità (Cliente, Azienda, Prodotto, Attività, Riparazione) deve essere presente una barra di ricerca che permette la ricerca secondo un qualsiasi campo
- Per ognuna delle entità (Cliente, Azienda, Prodotto, Attività, Riparazione) deve essere possibile ordinarle secondo un qualsiasi campo
- Visualizzare report su attività con grafico entrate/uscite per tipo di Attività
- Esportare in un foglio di calcolo i dati presenti nel gestionale

3.3 requisiti non funzionali

- Memorizzazione offline delle informazioni
- Grafica semplice e minimalista
- Prestazioni del sistema complessivo adatte ad un qualsiasi computer
- Disponibilità per il sistema operativo Windows

4. Descrizione Software

In base a tutti i requisiti raccolto durante la fasi precedente, il software che andremo a realizzare sarà il seguente:

Il software LC Elettronica sarà un gestionale incentrato sulla memorizzazione di clienti, aziende, prodotti, riparazioni ed attività. Nello specifico:

- Clienti, contiene i seguenti campi: Nome, Cognome, ID, Partita Iva, indirizzo, indirizzo spedizione, mail, PEC, codice fiscale, numero di telefono;
- Aziende, contiene i seguenti campi: Denominazione, partita iva, indirizzo, mail, PEC, codice identificativo, numero di telefono, indirizzo spedizione;
- Prodotti, contiene i seguenti campi: titolo, descrizione, quantità, prezzo vendita, prezzo acquisto, marca, modello;
- Attività, contiene i seguenti campi: descrizione, tipo, prezzo, segno, data;
- Riparazioni, contiene i seguenti campi: cliente, titolo, numero identificativo, data presa in carico, marca, modello, numero seriale, data acquisto, guasto, note;

Ognuna di queste sezioni conterrà tutte le operazioni di manipolazione delle informazioni, ossia: Inserimento, Cancellazione, Modifica, Visualizzazione, Importazione dati, Esportazione dati, inoltre per le seguenti classi sono state predisposte delle altre funzionalità:

★ Riparazioni:

Per questa sezione oltre le normali funzionalità è stata prevista la possibilità di stampare il pdf della riparazione con tutte le informazioni della riparazione e del cliente a cui è associata

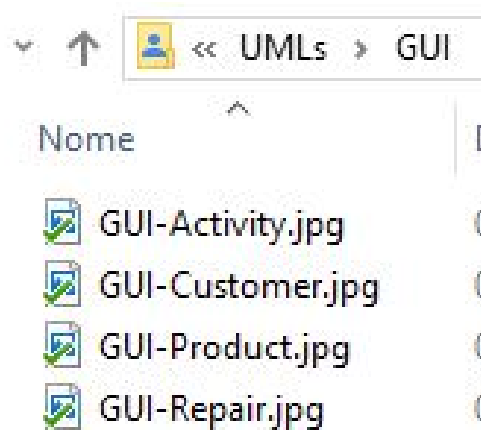
★ Attività:

Per questa sezione oltre le normali funzionalità è stata prevista la possibilità di visualizzare un rendimento mediante grafico di ogni attività, nello specifico il cliente richiede la possibilità data una particolare tipologia di attività di visualizzare il netto dato dalla somma in entrata di tutte le attività di tipologia uguale a quella presa in input meno le uscite.

E' doveroso fare una precisazione in merito al metodo di archiviazione da utilizzare per questo software, vista la scarsa copertura internet della zona dove situata l'azienda, a richiesta del cliente è necessario utilizzare un metodo di archiviazione offline, onde evitare perdite di informazioni e/o ritardi all'interno del software, è stato dunque previsto di archiviare tutte le informazioni del software all'interno di appositi file contenuti nella cartella principale del programma

5. UML

In allegato a questa relazione, è presente la cartella compressa UMLs.zip dove si troveranno i diagrammi UML delle classi suddivise per package. Si è scelto di suddividere il diagramma in sezioni per via della complessità e del numero di classi, che non permetteva una visione fluida in un singolo file.



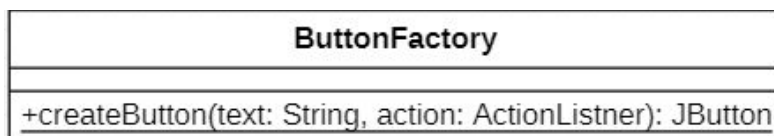
6 Design patterns utilizzati

Lo scopo principale del corso tenuto è quello di fornire competenze avanzate agli sviluppatori, non nello specifico come funzionalità di un particolare linguaggio o framework, ma piuttosto come progettazione logica del codice indipendentemente dal linguaggio utilizzato nel rispetto dei principi SOLID. A tal fine sono stati approfonditi 23 pattern, se consideriamo Simple Factory, Factory Method e Abstract Factory come pattern unico, suddivisi in tre categorie: creazionali, comportamentali e strutturali. Di questi 23, 5 sono stati utilizzati nella nostra applicazione.

6.1 Factory

Uno dei pattern creazionali più utilizzato è la Factory, suddivisa in tre versioni. La più semplice, consiste in una classe delegata della creazione degli oggetti. Una factory può essere invocata in svariati modi, tipicamente si chiama un metodo (anche static) che restituisce un'istanza dell'oggetto appena creato. In questo modo si astrae il meccanismo di creazione dell'oggetto, lasciando il client ignaro di come sia stato creato.

Lo scopo della nostra Simple Factory è quello di creare i bottoni dell'applicazione, definita come di seguito:



Il metodo *createButton*, ricevuti a parametro la stringa con il testo del bottone e l'evento al click, restituisce l'istanza del bottone da poter utilizzare nei frame/pannelli. Un esempio d'uso è la creazione del bottone per aggiungere un cliente:

```
//creiamo il bottone "Inserisci" che al click aggiunge un cliente in base al tipo
insert = ButtonFactory.createButton(SettingName.GUIINSERT, e -> {
    //{...}
});

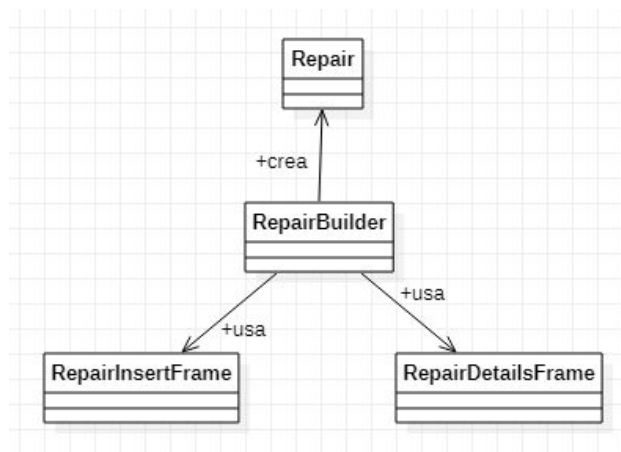
//aggiungiamo il bottone al pannello
buttonPanel.add(insert);
```

Nel contesto della nostra applicazione l'uso della factory è poco rilevante, essendo infatti delegata della semplice creazione di un bottone e dell'aggiunta di un ascoltatore. Tuttavia,

si è scelto di implementare questo meccanismo con questo design pattern perché se in uno sviluppo futuro si volessero cambiare alcuni settaggi comuni a tutti i bottoni, come ad esempio un *paintComponent* per stilizzarli, basterebbe modificare solo il metodo della factory.

6.2 Builder

Il pattern Builder è legato alla creazione di oggetti complessi. Siccome si è scelto di rappresentare gli elementi del nostro gestionale (clienti, riparazioni, prodotti, attività) non con una sequenza di attributi da inizializzare nel costruttore ma tramite una mappa di attributi, ci siamo avvalsi di un builder per settare tutte le entry della mappa.



Ad ogni metodo del builder è associata una chiave, che identifica quell'attributo all'interno della mappa.

Ad esempio, gli attributi di una riparazione sono così definiti:

```

public class Repair implements Item, java.io.Serializable {

    //{...}
    private HashMap<String, String> attributes = new LinkedHashMap();
  
```

La mappa degli attributi sarà nella forma ["chiave","valore"], ad esempio l'attributo *rottura* di un'istanza riparazione sarà una entry ["rottura","schermo con pixel bruciati"] nella mappa.

Nota: Si è scelta la sottoclasse **LinkedHashMap<K,V>** rispetto alla sua superclasse **HashMap** perché essa rispetta l'ordine delle chiavi (tipicamente quello di inserimento), requisito della renderizzazione della tabella, nella graphic user interface, contenente tutti gli elementi salvati.

Il relativo Builder:

```
public class RepairBuilder {
    private Repair repair;

    public RepairBuilder() { repair = new Repair(); }

    // {...}
    public void addBrake(String brake) { repair.add(SettingName.REPAIRBRAKE, brake); }
    // {...}
}
```

tramite l'invocazione del metodo *addBrake* aggiungerà una nuova entry agli attributi della riparazione (l'accesso alla mappa è oscurato dal metodo *add* della classe *Repair*).

Ad esempio, nel metodo che si occupa di inserire una riparazione si ha:

```
RepairBuilder builder = new RepairBuilder();
try {
    builder.addCustomer(comboBox.getItemAt(comboBox.getSelectedIndex()).getCustomer());
} catch (NullPointerException ex) {
    builder.addCustomer(null);
}
builder.addCode(code.getText());
builder.addNumber(number.getText());
builder.addPickUpDate(pickupdate.getJFormattedTextField().getText());
builder.addBrand(brand.getText());
builder.addModel(txtmodel.getText());
builder.addSerialNumber(serialnumber.getText());
builder.addPurchaseDate(purchasedate.getJFormattedTextField().getText());
builder.addBrake(brake.getText());
builder.addNote(note.getText());
try {
    builder.addPrice(Double.parseDouble(price.getText()));
} catch (NumberFormatException ex) {
    builder.addPrice(0);
}
builder.addPreventive(preventive.isSelected());

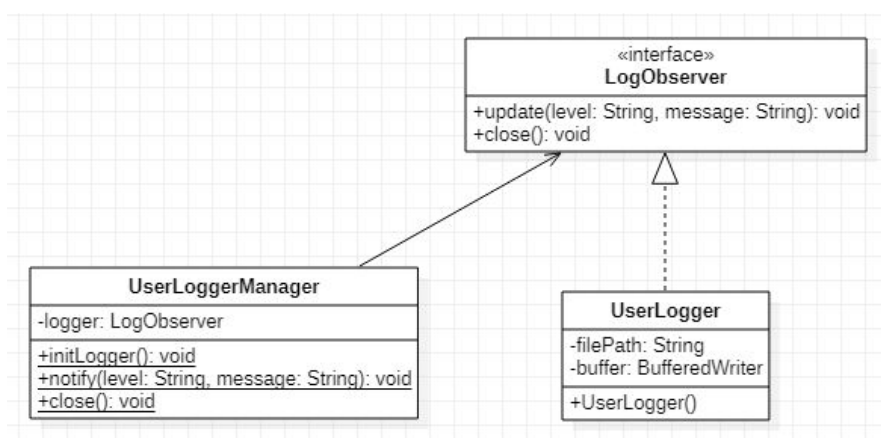
model.insert(builder.getResult());
```

Una volta creato l'oggetto *Repair*, verrà passato al metodo *insert* del *RepairTableModel* che si occuperà di aggiornare il manager e la tabella.

6.3 Observer

Alla base dell'architettura MVC (Model View Controller) il pattern Observer permette di notificare i cambiamenti di un osservato a tutti gli osservatori.

Nel nostro caso, l'osservato è il componente di logging, infatti ogni volta che il suo stato cambia (tramite l'invocazione del metodo *notify*) aggiorna l'osservatore che si occuperà di scrivere sul file di testo.



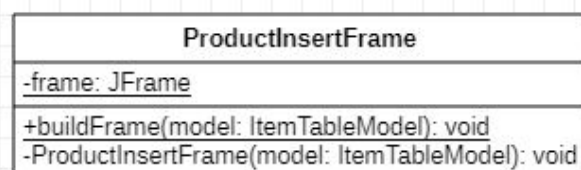
La classe *UserLoggerManager* tramite i suoi metodi statici può essere invocata da qualsiasi classe dell'applicazione ed aggiornerà tutti gli osservatori ad essa connessi.

6.4 Singleton

Questo semplice pattern, permette di creare istanze singole di una determinata classe. In questo modo, si può creare un solo oggetto e non vengono accettati duplicati.

Per far ciò si marca il costruttore **private** si fornisce un metodo statico che restituisce l'unica istanza disponibile.

Nel nostro caso si è implementata una versione leggermente diversa dalla definizione del pattern. Infatti, si volevano rendere alcune delle sottofinestre che l'applicazione creava esclusive, come ad esempio quella di inserimento. In modo che non si potessero avere contemporaneamente due finestre inserimento aperte. Per far ciò abbiamo marcato il costruttore **private** e creato un attributo *frame* statico.



Quando il client richiedeva un'istanza della finestra, se era già stata inizializzata la si chiudeva con il metodo `dispose()` e la si ri-inizializzava.

```
public class ProductInsertFrame extends JFrame {
    //frame singleton e il modello su cui desideriamo effettuare l'operazione di inserimento
    private static JFrame frame;

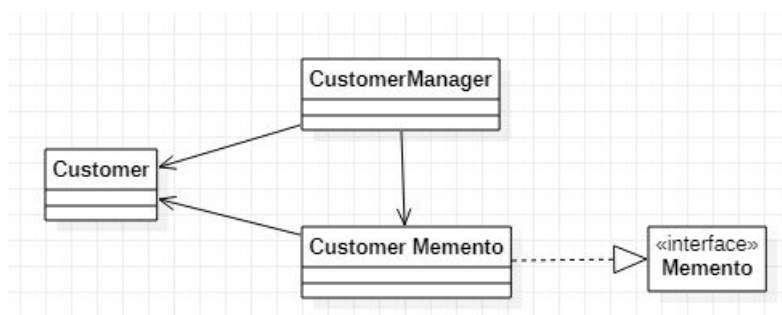
    //metodo static per creare un frame singleton, se è già presente lo elimina e lo ricrea
    //utile per evitare di aprire più finestre contemporaneamente
    public static void buildFrame(ItemTableModel model){
        if( frame == null){
            frame = new ProductInsertFrame(model);
        }else{
            frame.dispose();
            frame = new ProductInsertFrame(model);
        }
    }

    /**
     * Costruttore
     * @param model Il modello su cui inserire un prodotto
     */
    private ProductInsertFrame(ItemTableModel model){...}
}
```

6.5 Memento

Il pattern Memento serve per memorizzare lo stato di un oggetto, senza violare l'incapsulamento. Nel nostro caso si è voluta implementare l'annullamento di un'operazione di inserimento, rimozione o modifica.

Era quindi necessario salvare lo stato di un elemento prima di eseguire una delle operazioni CUD (Create, Upsert, Delete), e ripristinarlo nel caso in cui si volesse.



In ogni operazione CUD del Manager, si salvava lo stato ed il tipo di operazione.

```
@Override
public void create(Customer item) {
    saveState(item, SettingName.OPERATIONCREATE);
    customers.add(item);
}

public void saveState(Customer customer, String operation){
    this.prevCustomer = new CustomerMemento(customer, operation);
}
```

Dove *prevCustomer* è il nostro memento.

I caretaker, ovvero i tasti 'Annulla', ripristineranno l'operazione soltanto se è presente un memento salvato

```
//creiamo il tasto "Annulla operazione" che al click ripristina lo stato dei clienti all'operazione precedente
undoButton = ButtonFactory.createButton(SettingName.GUIUNDO, e -> {
    if(manager.getMemento().getState().getType().equals(SettingName.PERSONTYPE)){
        personModel.undo();
    }else{
        companyModel.undo();
    }
});
```

Essendo l'undo un'operazione che modifica la struttura del manager (e quindi della tabella visualizzata) è eseguita dalla relativa implementazione della classe astratta *ItemTableModel*, nel nostro caso *CustomerTableModel*:

```
//ricaviamo il memento
this.memento = manager.getMemento();

//in base al tipo di operazione eseguita la ripristiniamo
if(memento.getOperation().equals(SettingName.OPERATIONCREATE)){
    rows.removeIf(c -> c.getID() == this.memento.getState().getID());
    manager.restoreCreate();
}
```

che in base al tipo di operazione salvata nel memento, la ripristina.

Nell'esempio riportato si annulla un'operazione di inserimento eliminando dalla lista di clienti il cliente con ID uguale a quello salvato nel Memento.

7 Conclusioni

7.1 Commenti sul risultato finale

Dove è stato possibile e ritenuto opportuno sono stati applicati i design pattern, grazie ai quali è stato possibile ridurre l'agglomerato di classi e migliorare la qualità generale del codice. Inoltre siamo soddisfatti di aver rispettato i vincoli imposti dal cliente nel più breve tempo possibile

7.2 Possibili sviluppi futuri

Il software progettato soffre ancora oggi di alcune limitazioni che in futuro vorremmo risolvere, inoltre abbiamo intenzione di generalizzare il più possibile questo software, così da rendere più scalabile ed utilizzabile anche in contesti differenti. Mentre un altro aspetto che vorremmo curare in futuro è l'interfaccia grafica, utilizzando ad esempio javaFX.

7.3 Referenze

Il progetto è stato sviluppato e completato con l'ausilio di:

- Materiale didattico del corso tenuto dal prof. Gabriele Fici
- **Java Design Patterns: A Hands-On Experience with Real-World Examples**
- **StarUML** per la creazione dei diagrammi UML
- **JetBrains IDEA IntelliJ Community Edition** come IDE di sviluppo
- **JDatePicker** come libreria esterna per inserire le date
- **Apache PDFBox** come libreria esterna per creare PDF
- Protocollo GIT per lo sviluppo da remoto
- Google Drive come spazio cloud per condividere materiale inerente allo sviluppo