Documentazione

Benedetta Vitale - Emilio Meroni

13 gennaio 2024

Indice

Capitolo 1

Project Plan

1.1 Introduzione

Questo progetto verrà svolto da Benedetta Vitale ed Emilio Meroni, entrambi studenti al terzo anno d'ingegneria informatica presso l'università di Bergamo.

Il software che andremo a sviluppare(versione desktop) è pensato per dare supporto all'attività di ristorazione. In particolare, dovrà assistere alle mansioni dei camerieri, come ad esempio: prendere le ordinazioni, redigere il conto, trovare i tavoli disponibili, ecc.

Abbiamo scelto questa tipologia di sistema dato che Emilio lavora, nei week-end, presso un ristorante, e gli ha incuriosito la gestione interna tramite l'utilizzo dei palmari da parte dei camerieri.

1.2 Modello di Processo

Il modello di processo che seguiremo è quello della prototipazione, in particolare la prototipazione evolutiva [Figura: 1.1], nella quale si pensa di utilizzare circa dalle 4 alle 8 ore per effettuare un "giro". Questo processo è molto utile per quanto riguarda la costruzione dell'interfaccia grafica, dato che ci aiuterà a costruire più velocemente la GUI finale, tramite diversi prototipi d'inter-

facce utente. La parte principale della nostra applicazione, infatti, sarà la grafica.

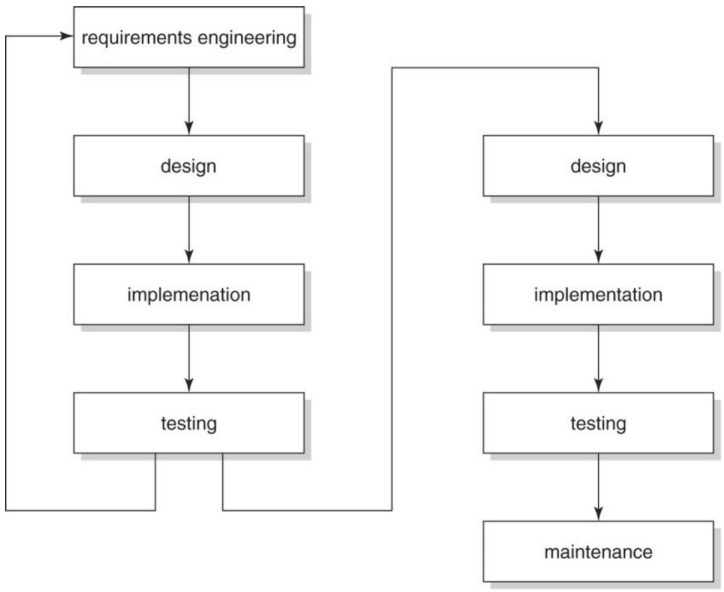

Figura 1.1: Modello Di Processo Evolutivo

1.3 Organizzazione del Progetto

Utilizzeremo una organizzazione a tre livelli:

1. Data Base
2. Logico
3. Presentazione

Il livello *Data Base*, come da nome, si occuperà della parte del DB, il quale sarà embedded.

Il livello *Logico*, possiamo posizionarlo graficamente tra il livello Data Base e quello Presentazione. Funge da intermediario tra i due livelli e fornisce gli oggetti principali per la gestione dell'applicazione.

L'ultimo livello, quello di *Presentazione*, sarà quello visto dall'utente che usufruirà dell'applicazione. Possiamo definirlo come il livello più esterno dove i dati verranno presentati in modo grafico e, la gestione dei tavoli e delle comande sarà fatta in modo interattivo.

Il progetto verrà suddiviso in:

- **Documentazione**; questa parte verrà svolta da entrambe le figure coinvolte nel progetto.
- **Progettazione design**; parte eseguita da Benedetta, che apprezza particolarmente questo ambito.
- **Codifica design**; questa porzione verrà eseguita principalmente da Emilio, dato che ha una maggiore esperienza sulla programmazione.
- **Codifica parte logica** e **data base**; le quali verranno scritte sia da Benedetta e sia da Emilio.

1.4 Standard, Linee guida, Procedure

Per la parte della stesura della documentazione si è scelto di utilizzare un tool molto utile nella scrittura di documenti professionali, *LaTeX*. Scelta quasi obbligata, derivata dall'utilizzo di *GitHub* insieme al programma di scrittura precedente *Microsoft Word*; il quale ha causato difficoltà nei "merge" su *GitHub*.

Mentre, per la parte di *codifica* abbiamo scelto di utilizzare lo standard definito da *Java*[1]. In primo luogo perché l'IDE utilizzato per la parte di programmazione è *Eclipse*, il quale fornisce strumenti per la formattazione e nominazione di metodi in modo automatico secondo gli standard di *Java*; inoltre, non avendo entrambi molta esperienza ci è venuto comodo utilizzare uno dei pochi standard di codificha che conosciamo.

Lo strumento che utilizzeremo per la condivisione della documentazione e del codice sarà, come già anticipato, la piattaforma di condivisione *GitHub*.

[1]Standard di Java si possono trovare su questo sito

1.5 Attività di Gestione

La priorità fissata per questo progetto sarà quella di avere una documentazione dettagliata, e, specialmente, di avere un project plan completo per l'inizio del mese di dicembre. Parellamente alla stesura di quest'ultimo si è deciso di dedicare, almeno una volta a settimana, del tempo sulla parte di progettazione.

1.6 Rischi

In questa sezione si discutono i rischi che potrebbero verificarsi durante lo sviluppo del progetto.

Un primo rischio, evidenziato da entrambe le parti, sono le difficoltà che potrebbero verificarsi nell'utilizzo di *windows builder* (plugin che consente la costruzione dell'interfaccia utente su *Eclipse*), dovuta a una bassa conoscenza del tool. Un probabile "effetto" sarà quello di allungare i tempi di codifica sul modulo di presentazione.

1.7 Personale

Il numero di persone che lavoreranno a questo progetto sarà molto ristretto, nello specifico sono:

- Benedetta Vitale
- Emilio Meroni

I quali parteciperanno in modo equo a quasi tutte le attività. Si è deciso che si lavorerà spesso in coppia, in particolar modo sulla parte di programmazione della GUI, dato che la parte grafica su *Java* non è mai stata approfondita da entrambe le parti, e questo progetto ha una parte di grafica molto importante.

1.8 Metodi e Tecniche

Per ogni nuova modifica aggiunta al progetto, prima di eseguire il "merge" su *GitHub*, la verificheremo tramite dei test. Sulla parte grafica il test verrà

eseguito in modo visivo, tramite l'esecuzione del programma. Se questo funzionerà ancora si procederà con l'aggiunta di nuove modifiche, così da avere sempre un codice eseguibile.

1.9 Garanzia e Qualità

Il programma dovrà essere utilizzato in ristoranti, e in particolare, dal personale che prenderà le ordinazioni. Quindi dovrà essere di *facile* utilizzo, con un focus maggiore sulla funzionalità che sull'estetica. Un'altra qualità che dovrà garantire è la *prevenzione di errori* da parte degli utenti, come ad esempio il "miss-click" (click errati o accidentali).

In particolare abbiamo evidenziato quattro qualità che il sistema dovrà possedere:

1. **Semplice**: Il programma sarà scritto per avere poche sezioni scritte e di facile comprensione, anche per chi si approccia al programma per la prima volta. Si utilizzeranno poche schermate che contengono tutto il necessario per le *macro* operazioni definite.

2. **Intuiblile**: L'utilizzo dei colori per indicare gli stati dei *tavoli*, in particolare si è deciso che: per i *tavoli liberi* si utilizzerà il colore verde, per i *tavoli occupati* il colore rosso e per i *tavoli da pulire* l'arancione [Figura: 1.2].

3. **Prevenzione Errori**: Per la prevenzione degli errori si utilizzeranno schermate *pop-up*, per la convalida degli input.

4. **Veloce**: Con questo termine si indica che il programma, che verrà utilizzato da dispositivi touch, dovrà essere principalmente composto da bottoni che minimizzano i tempi di utilizzo, puntando a un uso minimo della tastiera. Comportando un'esperienza più agevole per l'utente.

1.10 Workpackages

I moduli presenti in questo progetto, come già anticipato parzialmente nella sezione 1.3, saranno:

Figura 1.2: Esempio di colorazione dei bottoni per i tavoli

- **Documentazione**
- **Codifica design**
- **Codifica parte logica**
- **Codifica data base**

L'organizzazione dei pacchetti di lavoro, presenti su *GitHub*, sarà suddivisa in cartelle, per la parte di codifica raggrupperemo i diversi moduli in unico folder; mentre, la documentazione sarà posizionata in una cartella a parte.

Oltre ad avere queste directory, si avrà una parte dedicata ai modelli *ER*(riguardanti il data base) e una per la parte *UML* contenente tutti grafici per la progettazione.

1.11 Risorse

Gli strumenti che verranno adottati in questo progetto saranno:

- Per la parte di **codifica**, come già detto in precedenza, l'IDE *Eclipse*, uno strumento specifico per la scrittura di codice e gestione di progetti *Java*.
- In **scrittura** si utilizzerà l'editor di testo open source *VS Code*, con l'estensione *LaTeX Workshop*, per la scrittura di documenti in formato *LaTeX*. Contribuendo garantire una presentazione accurata dei documenti.
- Infine, riguardo agli strumenti di **Software Configuration Management**, adotteremo *GitHub*. Questa piattaforma sarà impiegata principalmente per la condivisione, gestione e tracciamento delle modifiche al software, oltre che per la documentazione.

1.12 Budget

Per quanto riguarda il budget, si è previsto un tempo totale di 80 ore lavorative. Con una forte attenzione sulla parte di documentazione (comprendendo anche la parte di documentazione del codice), rispetto a quella di codifica.

In prima approssimazione possiamo definire una divisione del tempo come mostrato in figura [1.3].

Figura 1.3: Diagramma di Gant

1.13 Cambiamenti

Per quanto riguarda le modifiche che verranno richieste si seguirà la seguente scaletta: inizialmente si verificherà la fattibilità, in caso negativo si cercheranno altre soluzioni che vadano incontro al cliente. Mentre, se si accettano, verrà creato un "branch" sulla piattaforma di *GitHub*. Da qui si aggiornerà la documentazione e i vari diagrammi *UML*, in seguito si procederà alle modifiche del codice. Infine prima di eseguire il "merge" delle modifiche con il *main branch*, si eseguiranno dei test di verifica della funzionalità e di correttezza.

1.14 Consegna

In questa sezione si discutono i metodi e le scadenze di consegna del progetto, in particolare la consegna si dividerà in due fasi:

- Consegna del **Project Plan**, il quale dovrà essere consegnato circa un mese prima del primo esame scritto, che si svolgerà nel mese di gennaio; quindi, per il mese di dicembre si dovrà effettuare la prima consegna.
- Consegna del **Progetto**, quest'ultimo avrà una scadenza più lunga, infatti, l'ultimo giorno di consegna sarà cinque giorni prima dell'esame orale.

Per quanto riguarda i metodi di consegna si dovrà condividere con il professore Gargantini la repository di *GitHub* contenente il progetto. Per la consegna della documentazione si dovrà indicare nel file *readMe*, della repository, la posizione del project plan. Mentre, per la consegna del progetto si dovrà creare un *issue*, intitolata "Approvazione Progetto" e assegnarla al professore.

Capitolo 2

Specifiche dei Requisiti

2.1 Introduzione

2.1.1 Obiettivo

L'obiettivo di questa applicazione sarà quello di diminuire i tempi di ordinazione aiutando il personale in sala con la gestione delle comande.

2.1.2 Scopo

Questa applicazione principalmente dovrà prendere ordinazioni e calcolare l'importo totale. Avere una organizzazione dei tavoli sapendo in qualsiasi momento lo stato di essi. Avendo anche la possibilità (a inizio o fine giornata) di modificare tutti i servizi che dispone per una maggiore organizzazione ai futuri cambiamenti che il ristorante adotterà all'interno di esso (es aggiunta di tavoli, di menù, ecc.)

2.1.3 Definizione del dizionario

- **Componenti**: sono sezioni di menù nei quali, al loro interno, contengono le diverse pietanze per quella specifica componente. Ad esempio: la pietanza "spaghetti alla carbonara" si trova all'interno della componente "primi".

- **Caposala**: persona che può accedere anche alle modifiche, cioè alle impostazioni, oltre alle funzioni classiche dell'applicazione.

- **Cameriere**: persona che può solo utilizzare l'app senza accedere alle impostazioni

2.2 Descrizione Globale Del Sistema

2.2.1 Funzioni del prodotto

In questa sezione descriviamo le funzioni che gli utenti potranno fare [Use case diagram a figura: 3.6]:

1. **Occupazione dei tavoli**: l'utente seleziona un tavolo libero e segnalerà il numero delle persone che si siederanno.

2. **Segnalazione Tavolo Da Pulire**: l'applicazione dovrà segnalare se un tavolo è ancora da pulire, e se verrà pulito, l'utente potrà indicare che il tavolo è pulito.

3. **Prendere le ordinazioni**: l'utente entrerà nel tavolo occupato e inizierà a prendere le varie ordinazioni scegliendo tra le diverse componenti. Potrà inserire commenti per ciascun piatto, aumentare la quantità per portata ed eliminare dei piatti aggiunti.

4. **Inviare gli ordini**: L'invio è il momento che renderà l'ordine immutabile e che lo renderà visibile alla cucina.

5. **Segnalare Pagamento effettuato**: l'utente entrerà nel tavolo occupato e, dopo che i clienti avranno pagato, potrà segnalare che il pagamento è stato effettuato.

6. **Impostazioni**: Questa sezione dovrà essere accessibile solo da dei caposala.

    - **Modificare la sala**: la modifica della sala comprende operazioni di modifica dei nomi e posti a sedere, l'eliminazione e l'aggiunta dei tavoli.
    - **Modifiche piatti e componenti**: Questa funzione servirà a modificare, aggiungere ed eliminare componenti e piatti:
        - **Piatti**: si considereranno modifiche sia sul nome sia sul prezzo
        - **Componenti**: Modifica del nome della componente

- **Modificare il prezzo del coperto**: si potrà modificare il prezzo che si desidererà per il coperto.

2.2.2 Caratteristiche dell'utente

Gli utenti saranno il cameriere e il caposala, i quali potranno accedere all'applicazione per assegnare tavoli, prendere ordinazioni e cambiare stato del tavolo. Solo il caposala, inoltre, potrà accedere alle sezioni delle impostazioni, nella quale potrà modificare la quantità dei tavoli, delle componenti, dei piatti e il costo del coperto.

2.2.3 Vincoli

Di seguito elenchiamo i vincoli che dovranno esserci:

- L'utente cameriere non potrà accedere alle impostazioni.

- Le caratteristiche di un tavolo potranno essere modificate solo quando il tavolo sarà nello stato di libero.

2.3 Requisiti specifici

2.3.1 Requisiti Interfacce utente

Generale

All'apertura si ha un interfaccia scorrevole con le diverse icone dei tavoli [Figure: 2.1 idealizzazione della grafia, 3.3 state machine], con il nome e il numero dei posti del tavolo. Il tavolo è colorato di rosso se è occupato, di verde se è libero e di arancione se deve essere pulito. In ogni schermata rimane sempre: in alto a destra i pulsanti "Impostazioni" e "Chiudi", in alto a sinistra un pulsante "Tavoli" per ritornare alla pagina principale dei tavoli.

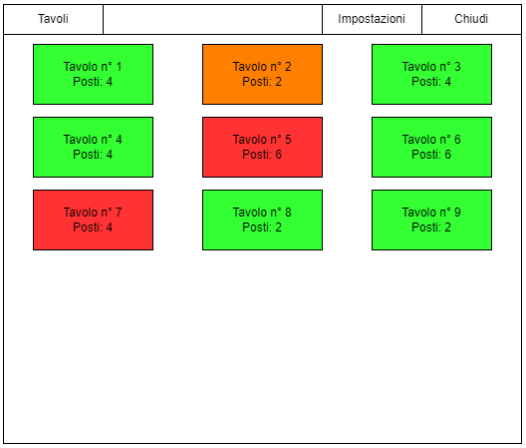

Figura 2.1: Vista dei Tavoli

Colorazione Tavoli

Se il tavolo è verde, il cameriere potrà schiacciarlo dove si aprirà una schermata pop-up in cui inserisce il numero di coperti, saranno due pulsanti e in messo un numero, in basso il "+" e in alto il "-", salvandolo tramite il pulsante "salva" il tavolo diventa rosso [Figura: 3.2 state machine].

Mentre se il tavolo che si è schiacciato è arancione si aprirà una schermata pop-up dove ci sarà un tasto con su scritto "pulito", che se schiacciato il tavolo diventerà da arancione a verde.

Al momento che le persone hanno pagato, il cameriere cliccherà, nel loro tavolo, il pulsante pagato (nella sezione ordinazione, figura: 2.3). Allora il tavolo diventa da rosso ad arancione [Figura: 3.1 state machine tavolo].

Ordinazione

Per i tavoli rossi, la schermata sarà divisa in due, dove la colonna a sinistra sarà molto più stretta rispetto a quella di destra [Figure: 2.3 idealizzazione grafica, 3.4 state machine].

Nella colonna di destra ci sarà la sezione del menù con relativa aggiunta dei piatti; questa colonna sarà a sua volta divisa in due righe, di cui quella in alto sarà molto più grande (circa 90%) rispetto a quella in basso (circa 10%). Quella di sopra sarà dedicata alla selezione dei menù, mentre quella di sotto è fissa ed è la sezione dei commenti.

Il menù avrà diversi pulsanti le componenti (figura: 2.3 in alto a destra). A seconda della componente selezionata si visualizzano i diversi piatti con i relativi prezzi. Dopo la selezione del piatto, a cui si può aggiungere un commento, si dovrà dar l'invio con il pulsante "Aggiungi" il quale si troverà accanto alla sezione commenti.

Schiacciato il tasto "aggiungi" il piatto apparirà nella colonna dei *piatti in ordinazione* (figura: 2.3 in alto a sinistra) e accanto al nome del piatto avrà i pulsanti "+" e "−", per aggiungere o diminuire la portata (se il cliente ha cambiato idea e non vuole più un piatto)[Figura: 2.2]. Il tasto "pagato" (figura: 2.3 in basso a sinistra) nel momento in cui si sta prendendo le ordinazioni non sarà possibile azionarlo, visto che questo tasto si potrà schiacciare dopo che tutte le ordinazioni siano state inviate tramite il pulsante "invia".

Figura 2.2: Vista Piatto Ordinato

Una volta inviato gli ordini si potranno vedere i piatti inviati nella sezione "resoconto"(figura: 2.3 in basso a sinistra). Inoltre, il tasto invia non sarà disponibile se non ci sono stati aggiunte all'ordine. In conclusione o è attivo il pulsante "invia" o è attivo il pulsante "pagato".

Se le pietanze ordinate sono tante, le parti di sinistra, quella alta e bassa, potranno scorrere e con esse i relativi pulsanti, "pagato" e "invia" (quindi per schiacciarli bisognerà scorrere fino in fondo, tale da visualizzare tutti gli ordini).

Impostazioni

Al click del pulsante impostazioni comparirà una schermata pop-up per effettuare il login[Figura: 2.4], se le credenziali saranno corrette verrà aperta la sezione delle impostazioni.

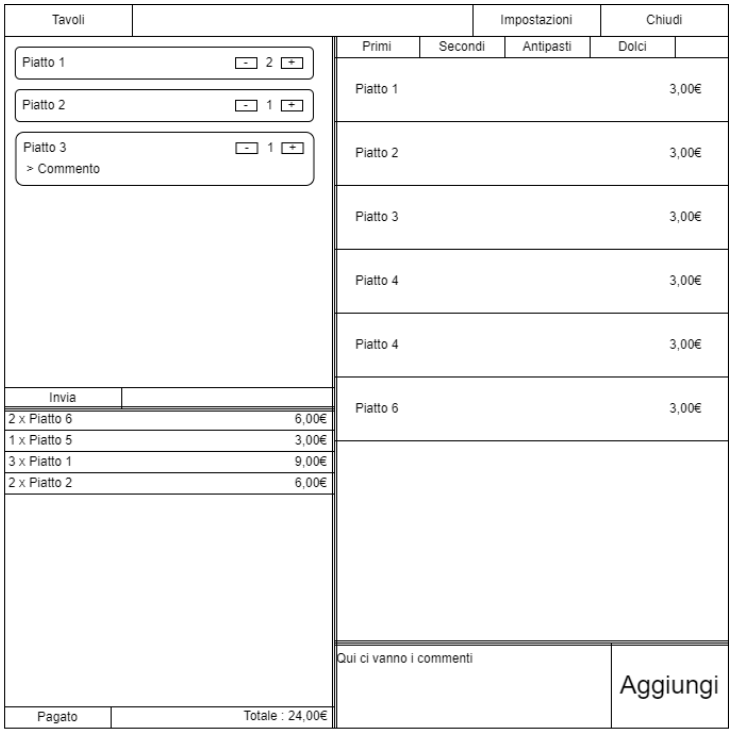

Figura 2.3: Vista delle ordinazioni

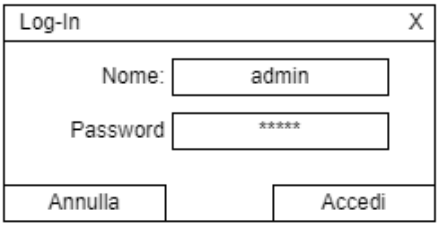

Figura 2.4: Vista Login

Quando si aprirà la schermata delle impostazioni si avranno quindi varie sezioni [Figura: 2.5 idealizzazione grafia, 3.5 state machine]. Una volta selezionata una sezione essa si amplierà.

La prima sezione sarà la sezione: "*Tavoli*". Aperta questa sezione al centro si visualizzano tutti i tavoli colorati in base al loro stato, sopra questa schermata ci sarà una scritta che indicherà quale tavolo si è selezionato,

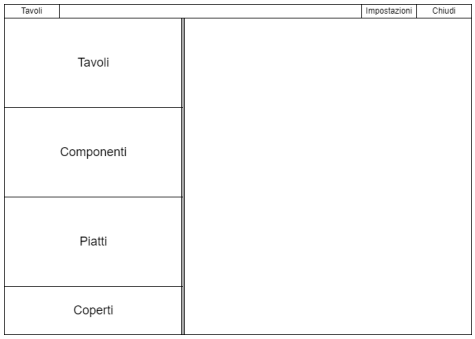

Figura 2.5: Vista delle impostazioni

mentre in fondo ci saranno tre pulsanti "*Aggiungi*", "*Modifica*" e "*Elimina*" [Figura: 2.6].

Figura 2.6: Vista delle impostazioni dei tavoli

Una volta cliccato il tasto Aggiungi o Modifica comparirà una schermata pop-up la quale chiederà il nome del tavolo e il numero di posti. Mentre, per il tasto elimina comparirà una schermata pop-up per la conferma dell'eliminazione.

Un'altra sezione sarà quella dedicata alle "Componenti" [Figura: 2.8]. Si avranno tre sezioni, In alto modifica/eliminazione, al centro la riordinazione mentre in basso l'aggiunta.

1. Nella sezione in alto avremo tutte le componenti, dopo averne selezionato una la si potrà modificare, tramite il pulsante "salva" si conferma la modifica, oppure eliminare, tramite il pulsante "elimina".

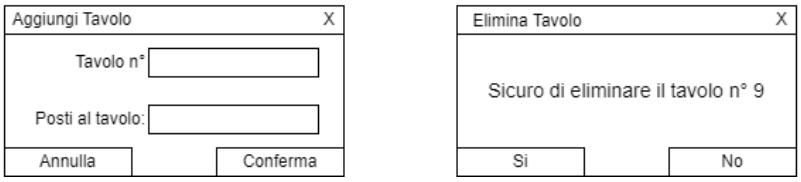

Figura 2.7: Pop-up impostazioni tavolo

2. Nella parte centrale ci sarà una tabella con la quale si potrà modificare l'ordine di visualizzazione delle componenti.

3. Mentre per la sezione aggiunta componenti avremo un campo in cui inseriremo il nuovo nome della componente, e sotto di esso, ci sarà un pulsante di conferma.

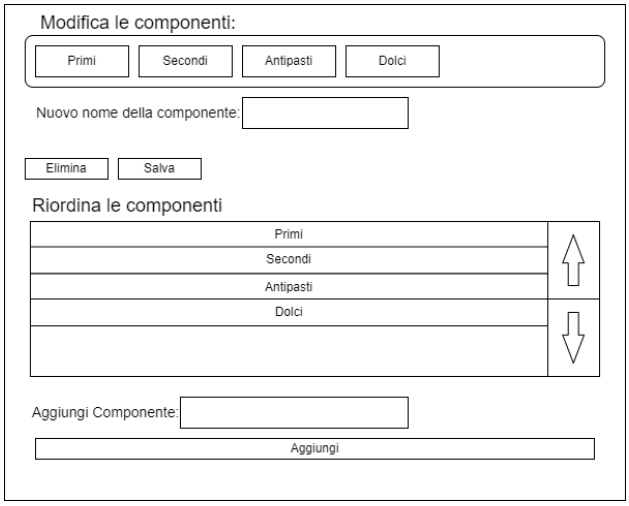

Figura 2.8: Impostazioni componenti

Un'altra sezione sarà la sezione "Piatti" [Figura: 2.9]. La quale tramite un selettore potremmo indicare quale componente visualizzare, e sotto di esso ci sarà una lista contenente tutti i piatti, con i relativi prezzi. Sotto ci saranno tre pulsanti: "Aggiungi", "Modifica" e "Elimina".

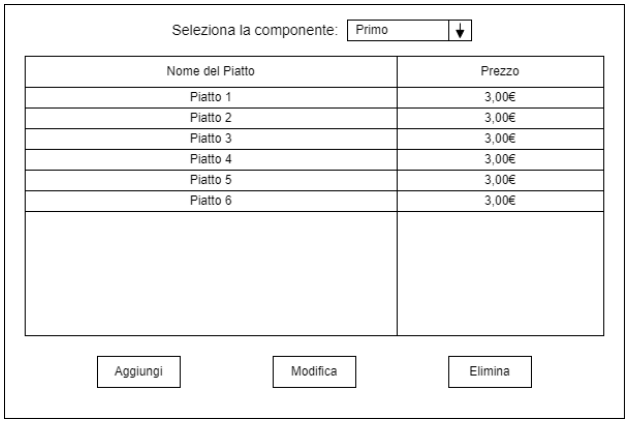

Figura 2.9: Impostazioni Piatti

Al click del pulsante “Elimina” comparirà una schermata pop-up nella quale chiederà la conferma dell’operazione. Mentre per i pulsanti “Aggiungi” e “Modifica” comparirà una schermata pop-up nella quale si potrà aggiungere il nome e il prezzo del piatto (in caso di modifica il nome e il prezzo visualizzati saranno quelli del piatto selezionato, nell’altro caso saranno vuoti).

Figura 2.10: Pop-up impostazioni piatti

Infine l’ultima piccola sezione è: “Modifica coperto” [Figura: 2.11]. Se schiacciata non apparirà una schermata come per le altre, ma si aprirà una schermata pop-up, nella quale si inserirà il nuovo valore che si confermerà tramite il pulsante "conferma", inoltre ci sarà anche il pulsante "annulla".

Figura 2.11: Pop-up Modifica Coperto

2.4 MoSCoW

Must have

Ordinazioni ai tavoli e la gestione degli stati dei tavoli.

Should have

Impostazioni, quindi modifiche: dei tavoli, delle componenti, dei piatti e del coperto.

Could have

Login caposala

Won't have

Gestione della concorrenza nella sezione impostazioni e nelle ordinazioni dei singoli tavoli

Capitolo 3

Diagrammi UML

Di seguito riportiamo i grafici UML.

3.1 State Machine

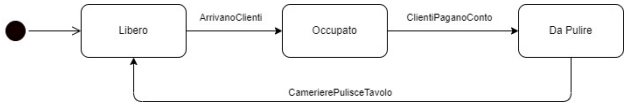

Figura 3.1: State machine tavolo

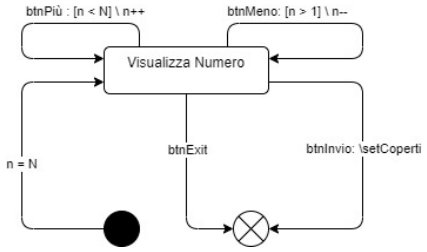

Figura 3.2: State machine grafica selezione del coperto

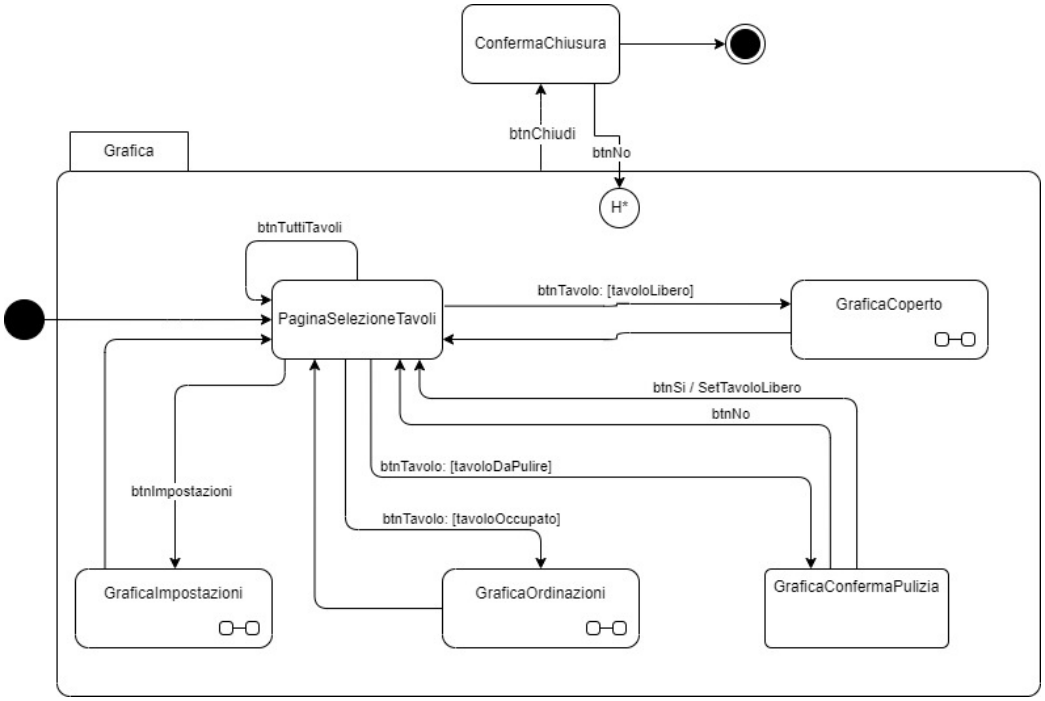

Figura 3.3: State machine grafica

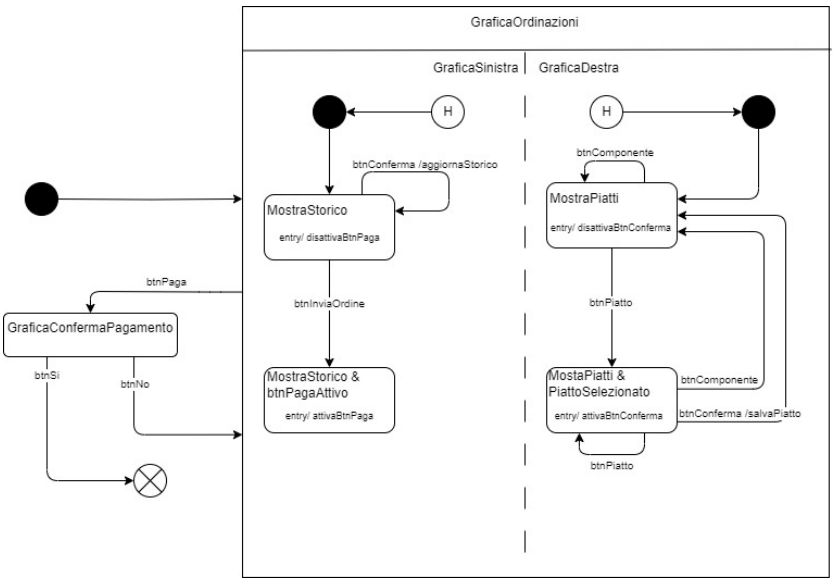

Figura 3.4: State machine grafica ordinazioni

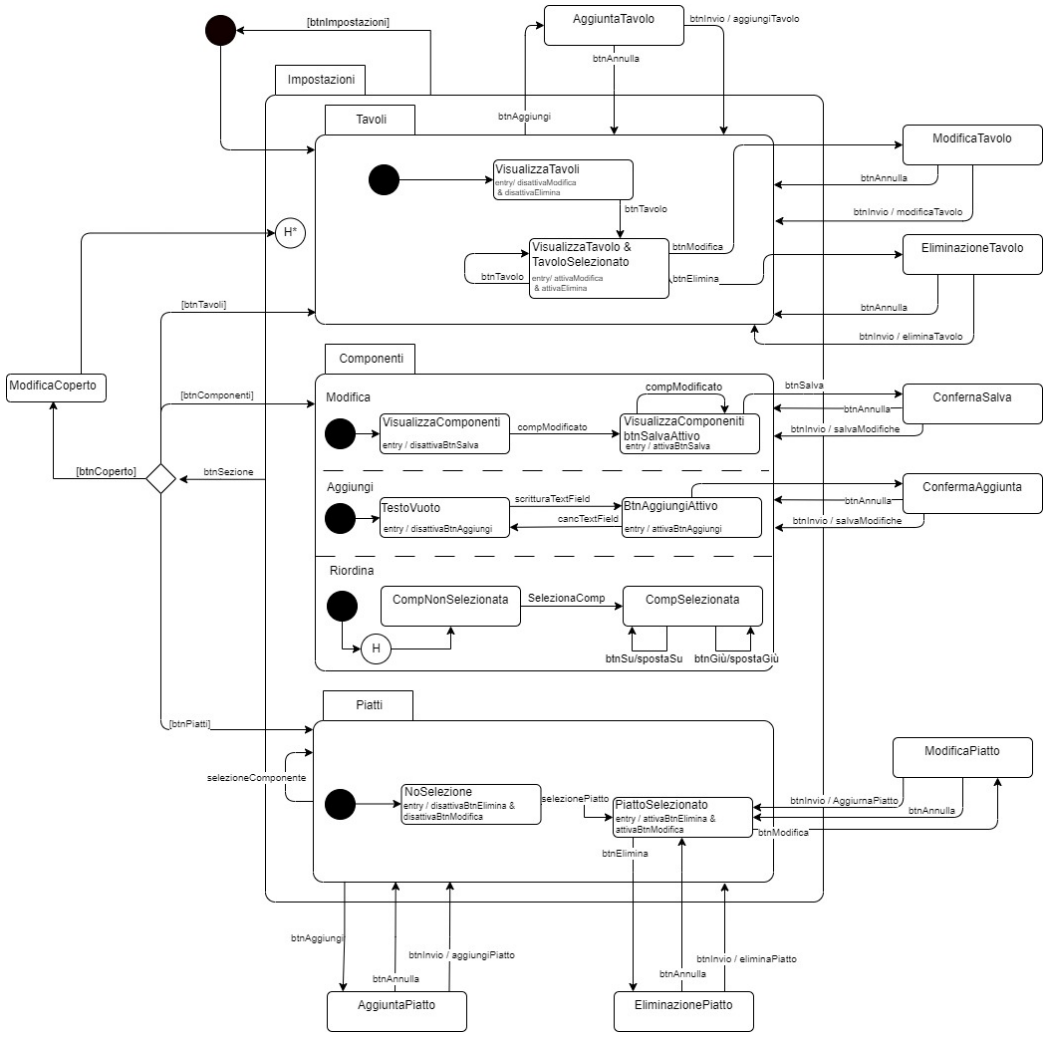

Figura 3.5: State machine grafica impostazioni

3.2 Casi D'Uso

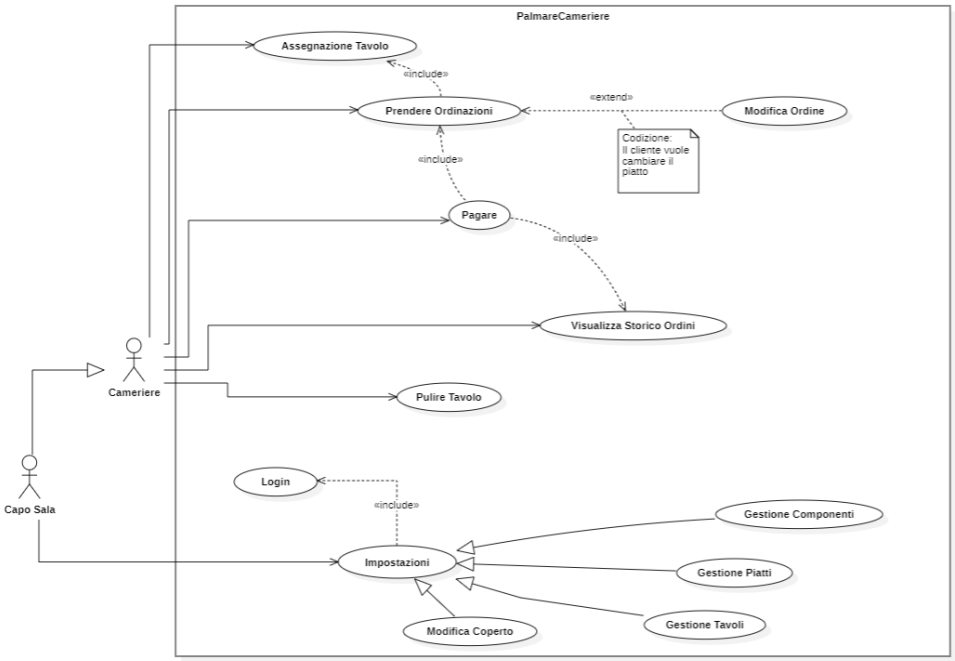

Figura 3.6: Use Case Diagram

3.3 Diagrammi di Sequenza

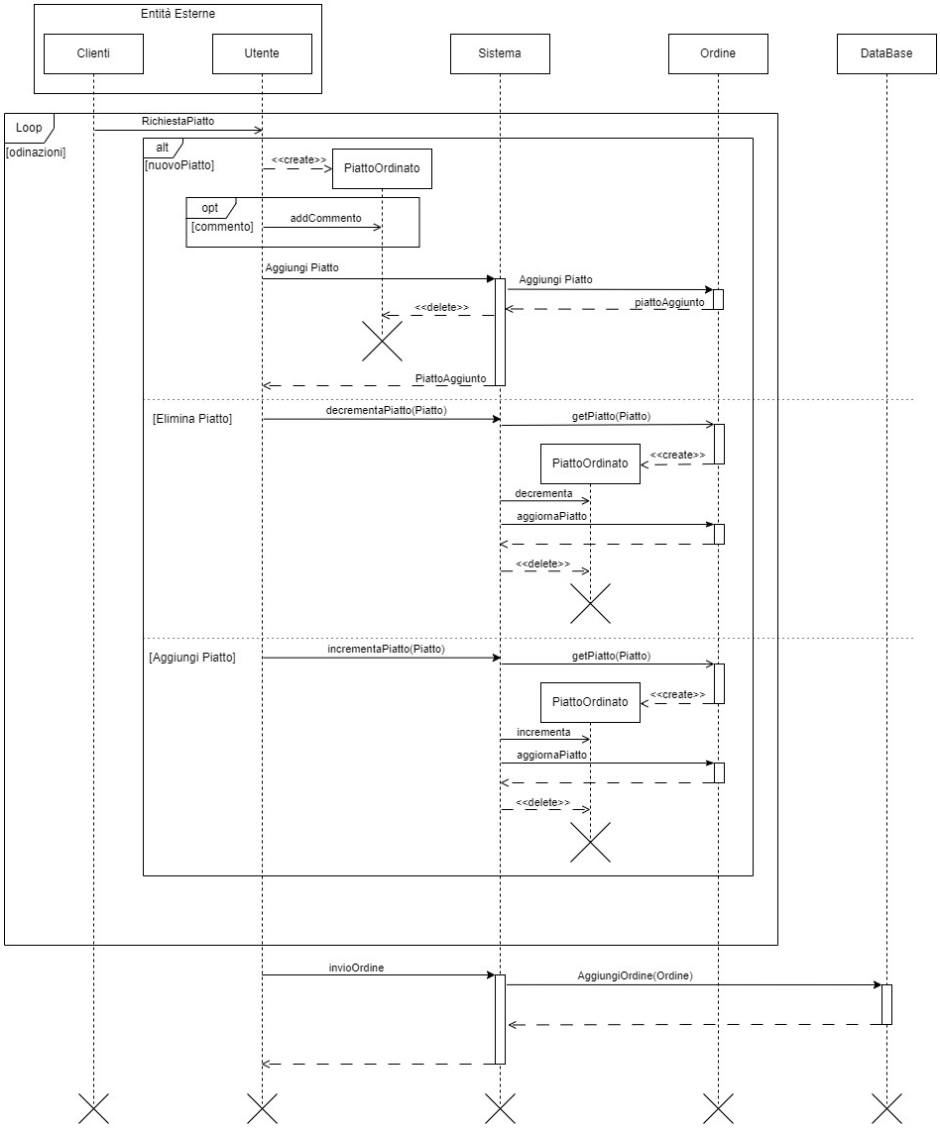

Figura 3.7: Diagramma delle sequenze ordinazione

3.4 Diagrammi delle Attività

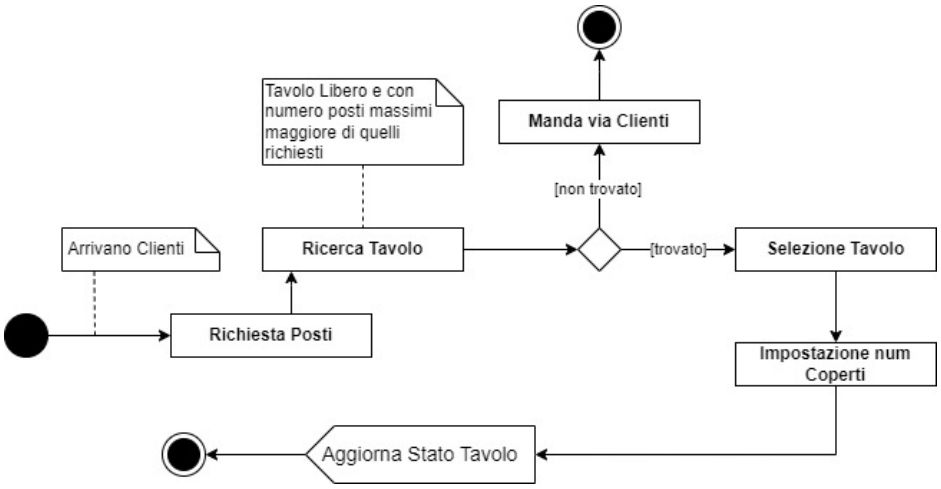

Figura 3.8: Diagramma attività arrivo clienti

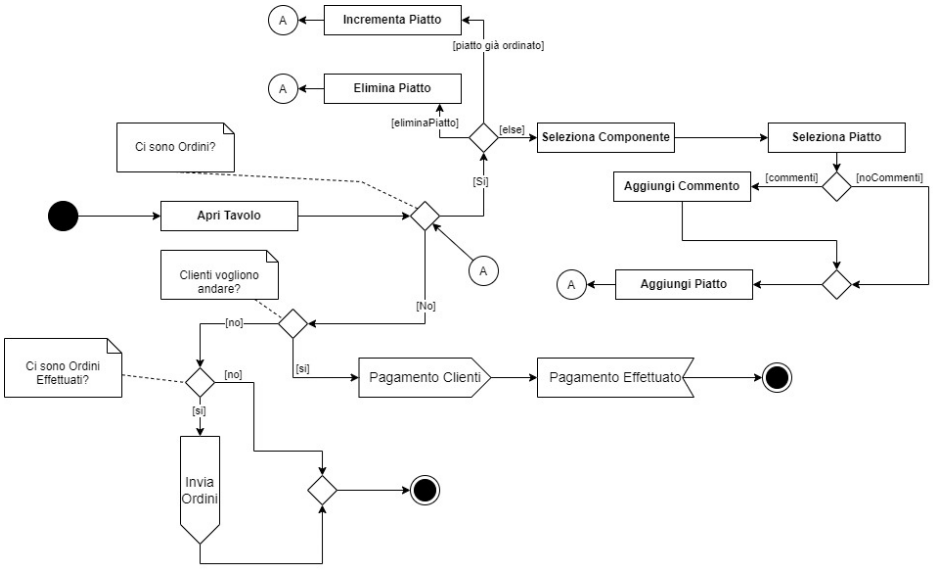

Figura 3.9: Diagramma attività ordinazioni

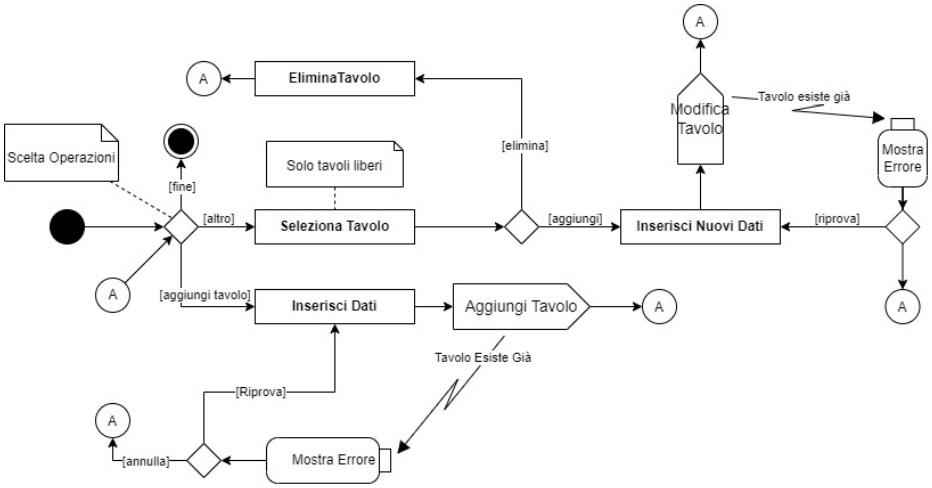

Figura 3.10: Diagramma attività impostazioni tavolo

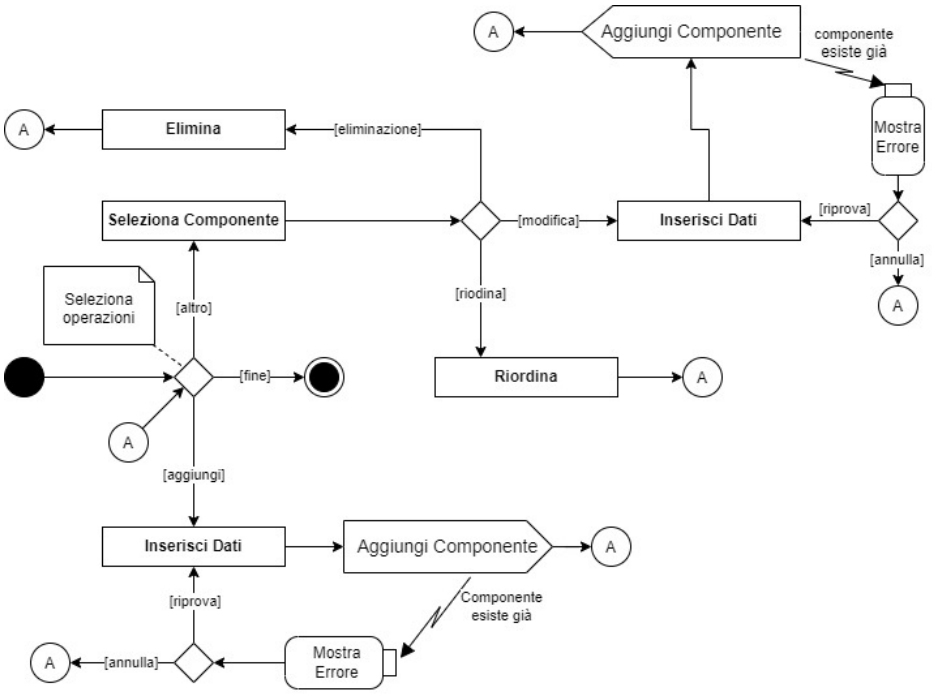

Figura 3.11: Diagramma attività impostazioni componenti

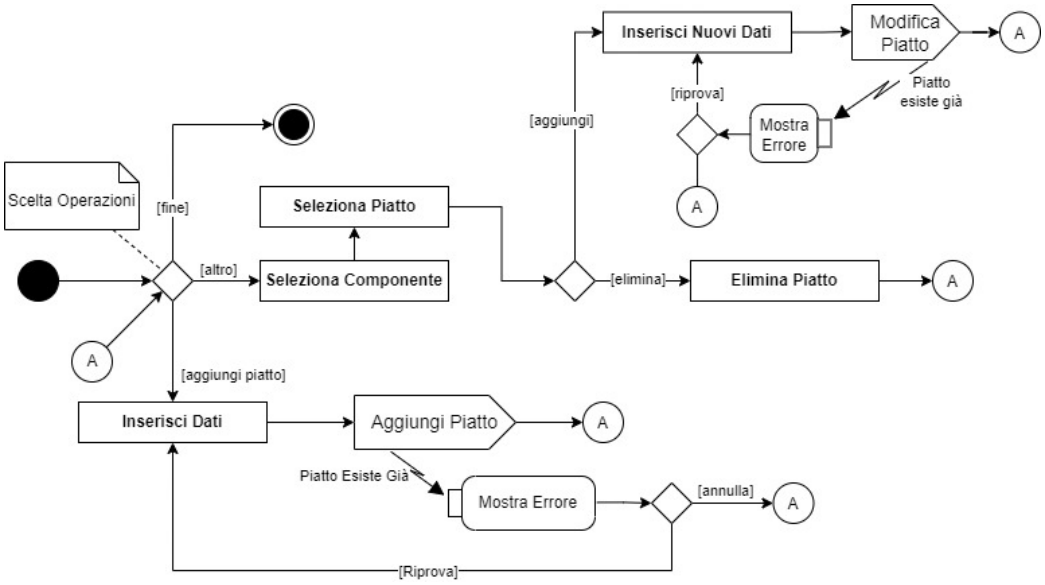

Figura 3.12: Diagramma attività impostazioni piatti

3.5 Diagramma ER

Di seguito riportiamo il diagramma ER per il data base

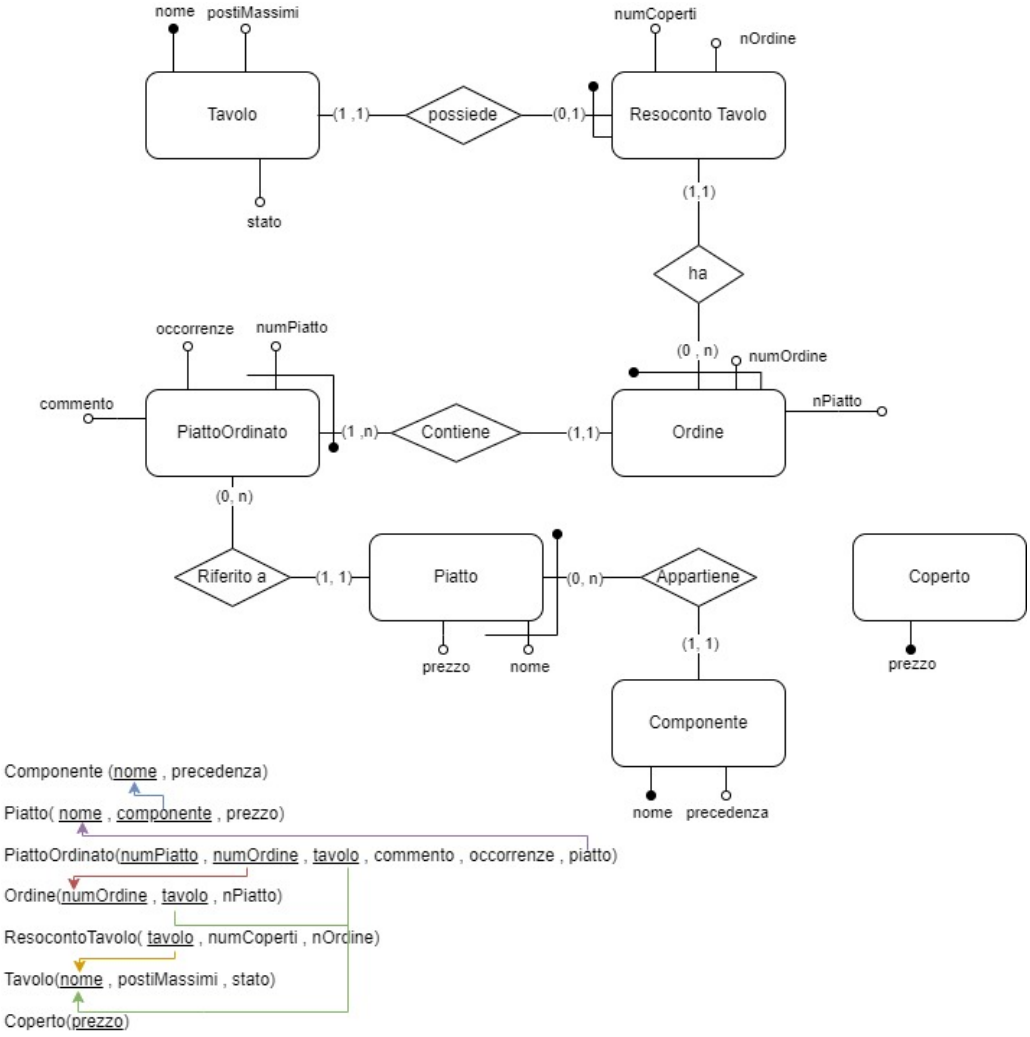

Figura 3.13: Diagramma ER

Capitolo 4

Analisi Parametri Codice

In questo capitolo ci occuperemo dello studio di alcuni parametri estratti grazie all'utilizzo del tool "CodeMR". Di seguito presenteremo 4 diversi parametri e li discuteremo, utilizzeremo la figura 4.1 come riferimento.

WMC: Weighted Method Count

"Conteggio Ponderato dei Metodi" questo parametro rappresenta la complessità di McCabe di una classe, esso è la somma ponderata dei diversi metodi in una classe, per esempio: *se consideriamo tutti i metodi di una classe a complessità pari a 1 allora WMC sarà pari alla somma dei metodi.*
Nel nostro caso abbiamo che il 50% ha un valore basso, il 36% medio-basso e il restante, in particolare la classe "DataService" medio-alto, quest'ultima, infatti, è la classe principale per l'estrazione, inserimento, modifiche ed eliminazione dei dati dal data base.

Coupling

Il secondo parametro che analizzeremo è l'accoppiamento. Quindi quanto una classe viene utilizzata al di fuori del proprio pacchetto. Nella maggior parte dei casi si ha un basso accoppiamento eccetto per una classe (sempre "DataService") la quale viene utilizzata molto dal modulo di presentazione, dato che serve per estrarre le informazioni dal data base.

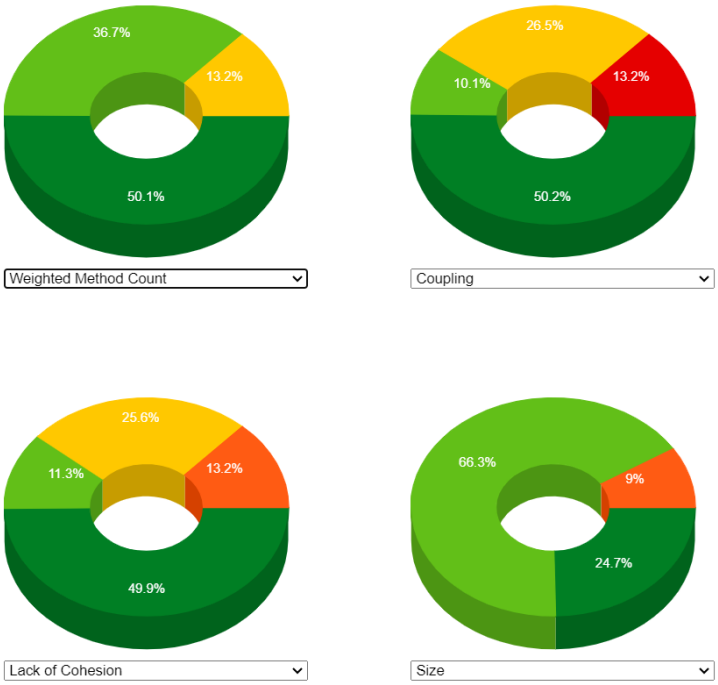

Figura 4.1: Grafici a torta dei diversi parametri

Lack of Cohesion

Un altro parametro è la "Mancanza di Coesione", in particolare se si ha *alta* coesione (utilizzo di una all'interno del proprio pacchetto) si avrà una *bassa* mancanza di coesione. Si può osservare che i parametri sono maggiormente di bassa/media-bassa, fattore molto buono il quale potrebbe indicare: robustezza, affidabilità e ri-utilizzabilità.

Size

Quest'ultimo parametro rappresenta le dimensioni di ciascuna classe, si può osservare dal grafico che maggiormente tutte le classi hanno una dimensione media-bassa, con il solo 9% media-alta dimensione.