

# *Progettazione e algoritmi*

*Progetto: concessionario intelligente*



Maffi Andrea, 1058173

Sassi Annamaria, 1063453

Testa Lorenzo, 1059562

Anno accademico 2021-2022

## Sommario

<i>Concessionario intelligente</i> .....	1
Iterazione 0.....	2
Analisi dei requisiti .....	2
Configurazione iniziale architettura.....	2
Modellazione casi d'uso.....	3
Architettura .....	5
Tool chain .....	5
Requisiti non funzionali .....	6
Iterazione 1.....	7
Casi d'uso.....	7
Diagramma UML di package.....	10
Diagramma UML delle componenti .....	11
Diagramma UML delle classi per definire data type classes .....	12
Diagramma UML delle classi per definire le interfacce .....	13
Pseudocodice e analisi complessità tempo.....	13
Analisi statica.....	14
Analisi dinamica .....	15
Iterazione intermedia.....	16
Analisi requisiti e casi d'uso .....	16
Design dell'architettura software .....	16
Analisi statica.....	17
Analisi dinamica .....	17
Iterazione finale.....	18
Analisi requisiti e casi d'uso .....	18
Design dell'architettura software .....	21

Analisi statica.....	21
Analisi dinamica .....	22
<i>Guida all'installazione dell'applicazione</i> .....	24
<i>Sviluppi futuri</i> .....	25
<i>Uso dell'applicazione</i> .....	26
Interfaccia meccanico .....	26
Interfaccia venditore .....	27
Interfaccia capo officina.....	27

# Concessionario intelligente

In una concessionaria con diverse filiali sparse sul territorio, le auto usate possono essere vendute nella filiale in cui si trovano o in un'un'altra. Le filiali hanno in comune un database contenente tutte le macchine usate presenti.

Una volta venduta una macchina, l'incaricato post-vendita determina tutti gli interventi che devono essere eseguiti e solo in seguito a ciò viene assegnato lo slot di lavorazione tramite l'algoritmo. Questi interventi possono essere effettuati nella filiale in cui si trova l'auto oppure in una qualsiasi altra filiale, a seconda del metodo più efficiente (tenendo anche conto del tempo di trasporto tra le filiali).

Gli interventi possono essere modificati, cancellati oppure aggiunti; in tutti e tre i casi viene aggiornato lo slot di lavorazione e viene modificato il carico dell'officina.

Il venditore può anche inserire nel database nuove macchine in vendita.

Il capo officina visualizza tutti gli interventi che ha in carico la filiale e quindi può assegnare i singoli interventi ai meccanici.

I meccanici possono visualizzare le lavorazioni che gli sono state assegnate e una volta terminate devono segnalarlo tramite l'apposita interfaccia.

Il capo filiale può controllare il database contenente le macchine vendute (e relativi interventi) e quelle in vendita. Un altro compito a lui assegnato è quello di aggiungere nuovi ponti, nel caso in cui la concessionaria abbia un carico di lavoro superiore rispetto a quello che può sostenere.

Nella filiale centrale è presente il capo di tutte le filiali e può controllare le macchine in transito tra le filiali, quelle vendute e quelle in vendita. Inoltre, può aggiungere le nuove filiali.

Nella filiale centrale è contenuto il database centrale e l'algoritmo.

L'algoritmo prende in considerazione la capacità di lavoro di ogni officina e la lista degli interventi delle singole auto.

La filiale centrale deve poter vedere tutte le giacenze e, sulla base del numero delle auto in officina e dei tempi necessari al trasporto delle auto, decidere come smistare il carico di auto in attesa.

La filiale centrale conosce tutti i tempi di trasporto tra le filiali.

Il capo officina cambia lo stato delle auto.

Quando l'auto viene ritirata dal concessionario, il proprietario diventa il concessionario stesso, questa informazione viene riportata anche sul database.

## *Iterazione 0*

### Analisi dei requisiti

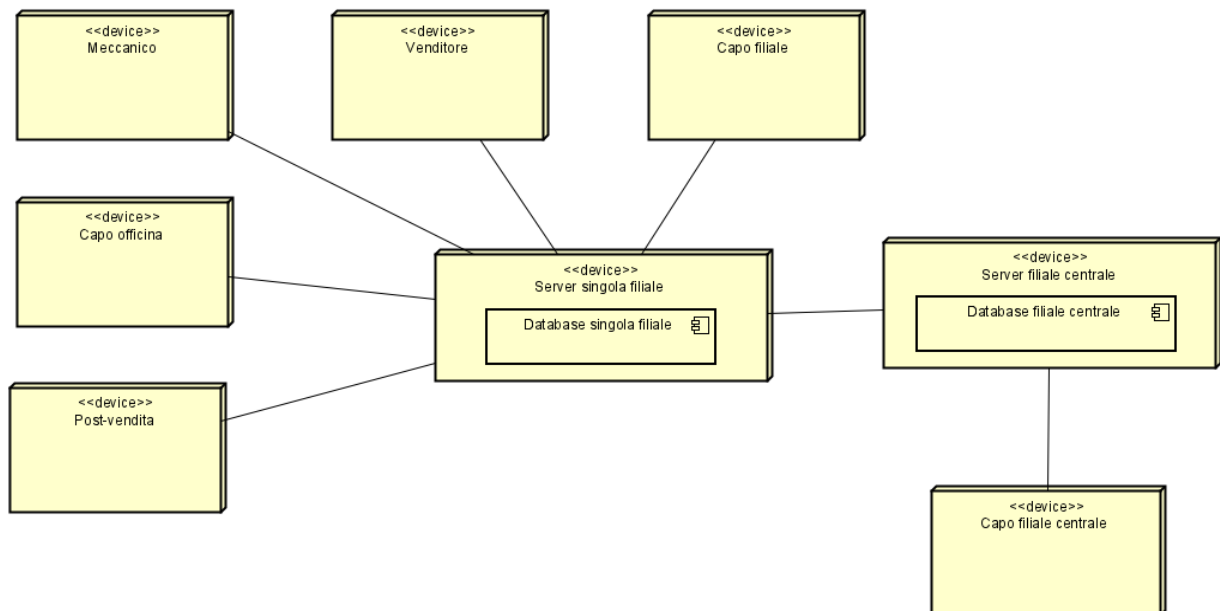
Il sistema è composto da:

1. Una serie di interfacce grafiche: una per ogni tipologia di utente che può usare il sistema, in particolare quella del meccanico, del capo officina, del venditore, del post-vendita e del capo filiale.
2. L'algoritmo, grazie al quale è possibile determinare la schedulazione delle macchine vendute in modo che siano riparate nel più breve tempo possibile.
3. I moduli della struttura a microservizi.

### Configurazione iniziale architettura

Si è pensato a un'architettura nella quale ogni sottosistema rappresenta una tipologia di utente e questi interagiscono con il database della singola filiale.

Dato che ci sono più filiali, si è individuata una filiale centrale nella quale è presente il server centrale e il capo filiale centrale; quest'ultimo può controllare le altre filiali e quindi può anche aggiungere nuove filiali.



*Figure 1: Architettura di base*

## Modellazione casi d'uso

I casi d'uso sono stati rappresentati attraverso un grafico e poi anche in forma tabellare per raggrupparli a seconda del tipo di utente che li può eseguire.

Gli attori presenti sono: capo officina, post-vendita, venditore, meccanico, capo filiale e capo filiale centrale.

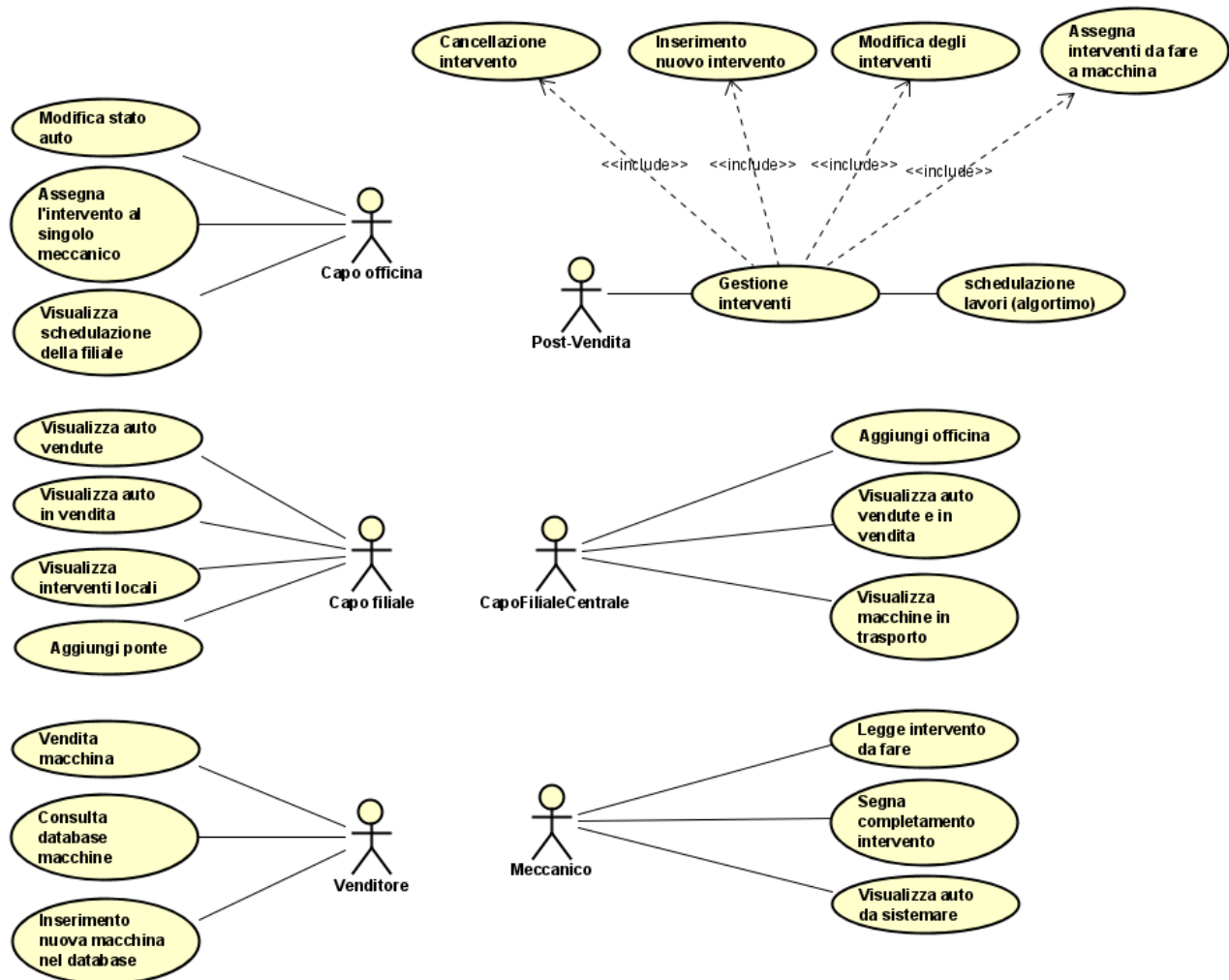


Figure 2: Iterazione 0 - Casi d'uso

Il nome usato per ogni caso d'uso è un'abbreviazione dell'attore che può eseguire le azioni descritte.

Nome: PV

Attori: Post-vendita

Eventi:

1. Gestione degli interventi (eliminazione, modificazione, aggiunta, assegnamento intervento a macchina)

Nome: V

Attori: Venditore

Eventi:

1. Visualizzare database delle macchine in vendita
2. Vendere una macchina
3. Inserisce nuova macchina nel database

Nome: CO

Attori: Capo officina

Eventi:

1. Visualizza la schedulazione della sua filiale
2. Assegna gli interventi ai meccanici
3. Modifica stato delle auto (in coda, lavorazione, trasporto alla filiale di lavorazione e trasporto alla filiale di consegna)

Nome: CF

Attori: Capo filiale

Eventi:

1. Visualizza le macchine vendute
2. Visualizza le macchine in vendita
3. Inserisce un nuovo ponte
4. Visualizza gli interventi

Nome: M

Attori: Meccanico

Eventi:

1. Visualizza le macchine da sistemare (relative al suo ponte)
2. Visualizza gli interventi da effettuare
3. Segnalazione fine intervento

Nome: CFC

Attori: Capo filiale centrale

Eventi:

1. Visualizza le macchine in transito
2. Visualizza le macchine vendute e in vendita
3. Inserisce nuova officina

## Architettura

Questo progetto è stato sviluppato attraverso un'architettura a microservizi, cioè il codice è distribuito in parti piccole e gestibili (componenti) e ogni servizio viene creato in modo indipendente dagli altri servizi. I componenti indipendenti interagiscono e comunicano tramite contratti prestabiliti (API). Grazie a questo tipo di struttura i componenti risultano coesi al loro interno e poco accoppiati tra di loro.

Le risorse vengono suddivise in differenti layer:

1. Repository layer: strato dell'applicazione più vicino alle risorse e si interfaccia con il database.
2. Service layer: funge da interfaccia tra gli strati Repository e Web; i componenti service vengono utilizzati per la gestione e la trasformazione di un dato ottenuto dal database per la sua rappresentazione all'esterno.
3. Controller: strato più pubblico e servono a definire le varie URI (Uniform Resource Identifier) che identificano unicamente una risorsa e sulle quali sono assegnati i metodi http adatti all'operazione da svolgere.
4. Domain: contiene le entità per la rappresentazione e lo scambio di dati.

## Tool chain

Attività	Tool
Modellazione UML	Astah UML
Linguaggio di programmazione	Java
Framework backend	Spring Boot
DBMS	MySQL
IDE	Eclipse
GUI	WindowBuilder
Analisi statica	STAN4J
Analisi dinamica	JUnit 4
Versioning	GitHub
Programmazione condivisa	Code together in eclipse



## Requisiti non funzionali

Grazie all'architettura a microservizi è stato possibile considerare alcuni specifici requisiti non funzionali:

1. **Manutenibilità:** è più facile individuare possibili correzioni e quindi poter apportare modifiche nell'unico punto interessato senza dover cambiare anche gli altri moduli.
2. **Scalabilità:** è possibile aggiungere nuove funzionalità alle diverse componenti e aggiungere nuove interfacce senza intaccare il corretto funzionamento del sistema e senza dover modificare tutti i moduli.
3. **Affidabilità:** la schedulazione che viene effettuata viene salvata in modo che non venga persa.

## Iterazione 1

In questa iterazione abbiamo sviluppato in modo più dettagliato i casi d'uso e la struttura dell'architettura. Inoltre, abbiamo iniziato a predisporre l'ambiente di Eclipse e a creare la struttura del progetto.

### Casi d'uso

Rispetto all'iterazione precedente è stato aggiunto l'amministratore di rete che può inserire nuovi utenti, eliminare account o modificare i dati di un utente. Ogni operatore, di conseguenza, prima di poter effettuare una qualsiasi operazione deve effettuare il login in modo che il sistema, conoscendo la mansione dell'utente, possa limitarne il comportamento.

Le interfacce, di conseguenza, non saranno uguali per tutti gli utenti, ma dipendono dalla tipologia della mansione.

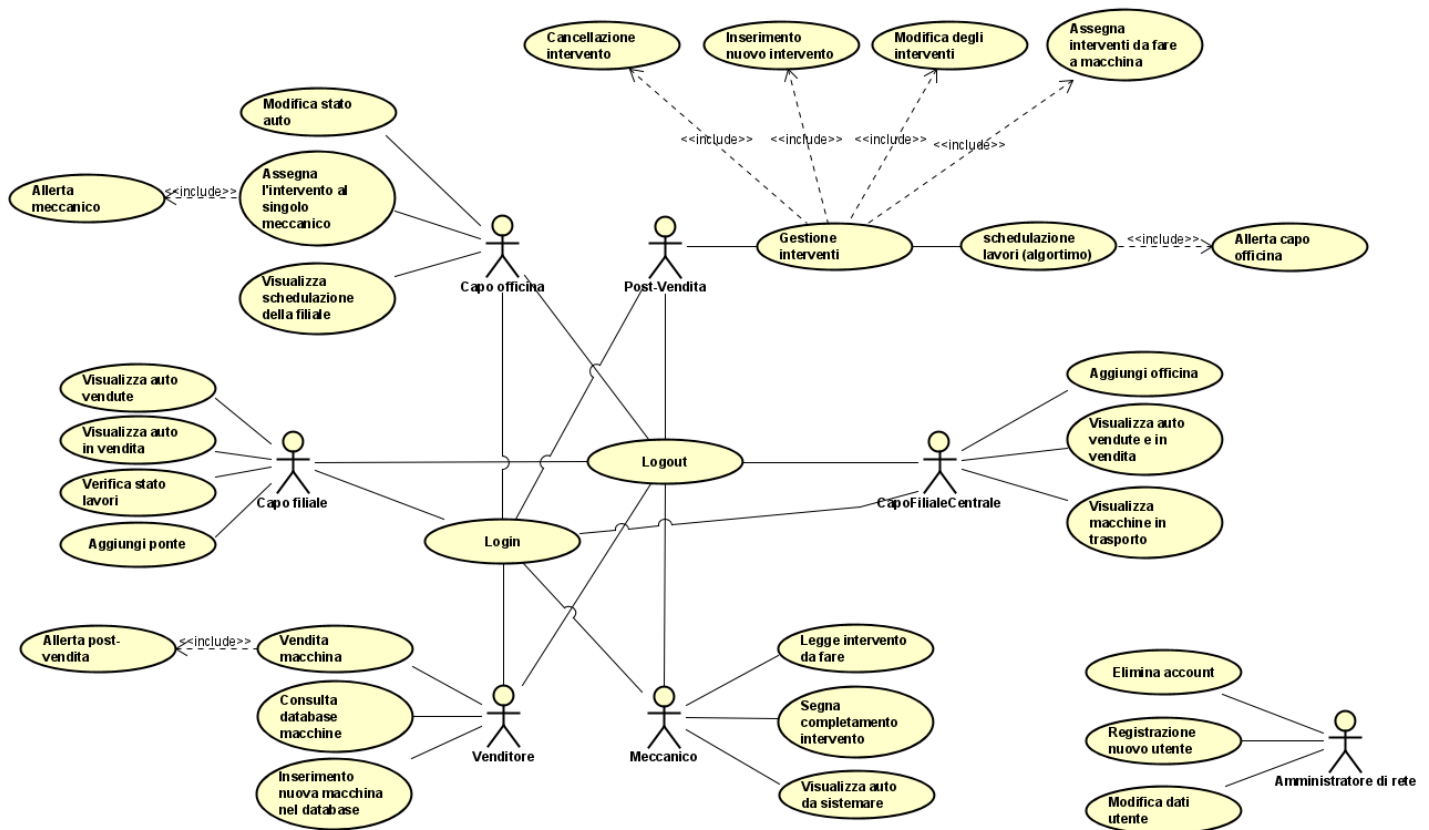


Figure 3: Iterazione 1 - Casi d'uso

Casi d'uso rappresentati sotto forma di tabella:

Nome: PV

Attori: post-vendita

Precondizioni: login

Eventi:

1. Gestione degli interventi
  - a. Cancellazione intervento
  - b. Inserimento nuovo intervento
  - c. Modifica degli interventi
  - d. Assegnare intervento a una macchina

Postcondizioni: logout

Nome: V

Attori: Venditore

Precondizioni: login

Eventi:

1. Visualizzare database delle macchine in vendita
2. Vendere una macchina
3. Inserisce nuova macchina nel database

Postcondizioni: logout

Nome: CO

Attori: Capo officina

Precondizioni: login

Eventi:

1. Visualizza la schedulazione della sua filiale
2. Assegna gli interventi ai meccanici
3. Cambia stato delle auto (in coda, lavorazione, trasporto alla filiale di lavorazione e trasporto alla filiale di consegna)

Postcondizioni: logout

Nome: CF

Attori: Capo filiale

Precondizioni: login

Eventi:

1. Visualizza le macchine vendute
2. Visualizza le macchine in vendita
3. Inserisce un nuovo ponte
4. Visualizza gli interventi

Postcondizioni: logout

Nome: M

Attori: Meccanico

Precondizioni: login

Eventi:

1. Visualizza le macchine da sistemare (relative al suo ponte)
2. Visualizza gli interventi da effettuare
3. Segnalazione fine intervento

Postcondizioni: logout

Nome: CFC

Attori: Capo filiale centrale

Precondizioni: login

Eventi:

1. Visualizza le macchine in transito
2. Visualizza le macchine vendute e in vendita
3. Inserisce nuova officina

Postcondizioni: logout

Nome: AR

Attori: Amministratore di rete

Precondizioni: login

Eventi:

1. Inserisce nuovi utenti
2. Elimina utenti
3. Modifica i dati degli utenti

Postcondizioni: logout

Si può osservare che i casi d'uso si dividono in quattro macroaree: interventi, automobile, account manager, officina, quindi possiamo suddividerli come segue.

<p>Interventi:</p> <ol style="list-style-type: none"><li>1. Gestione interventi<ol style="list-style-type: none"><li>a. Assegnamento interventi alla singola macchina</li><li>b. Modifica di un intervento</li><li>c. Inserimento di un intervento</li><li>d. Cancellazione di un intervento</li></ol></li><li>2. Visualizzazione degli interventi programmati in una determinata filiale</li><li>3. Assegnazione interventi ai meccanici</li><li>4. Segnalazione del completamento dell'intervento</li></ol>
<p>Auto Manager:</p> <ol style="list-style-type: none"><li>1. Visualizzare database contenente le macchine in vendita</li><li>2. Vendere una macchina</li><li>3. Inserimento nuova macchina in vendita nel database</li><li>4. Visualizzazione database contenente macchine vendute</li><li>5. Visualizzazione database contenente macchine in trasporto</li></ol>
<p>Account Manager:</p> <ol style="list-style-type: none"><li>1. Login</li><li>2. Logout</li><li>3. Registrazione nuovo utente</li><li>4. Eliminazione account</li><li>5. Modifica dati utente</li></ol>
<p>Officina:</p> <ol style="list-style-type: none"><li>1. Aggiungere un'officina</li><li>2. Aggiungere un ponte</li></ol>

## Diagramma UML di package

I casi d'uso sono raggruppati in quattro macro-gruppi, per questo motivo abbiamo sviluppato quattro package indipendenti all'interno del package *Concessionario Intelligente*.

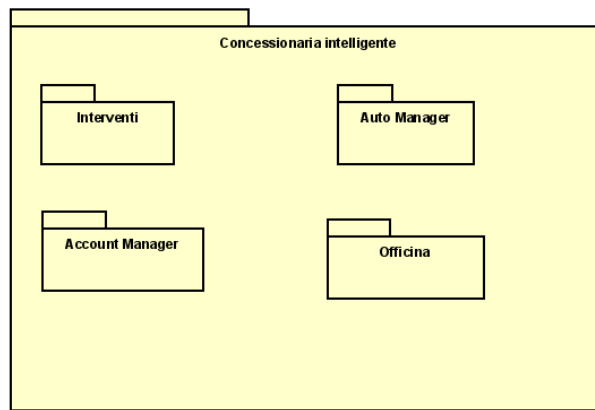


Figure 4: Iterazione 1 - Diagramma UML di package

## Diagramma UML delle componenti

Dato che il componente *account manager* serve per gestire i login, esso fornirà il servizio per accedere al sistema. Tutti gli altri componenti, invece, posso visualizzare e modificare il database grazie a dei servizi appositi.

Il componente *interventi* ha bisogno anche delle informazioni relative alle macchine per poter associare ad ogni lavoro un'auto specifica.

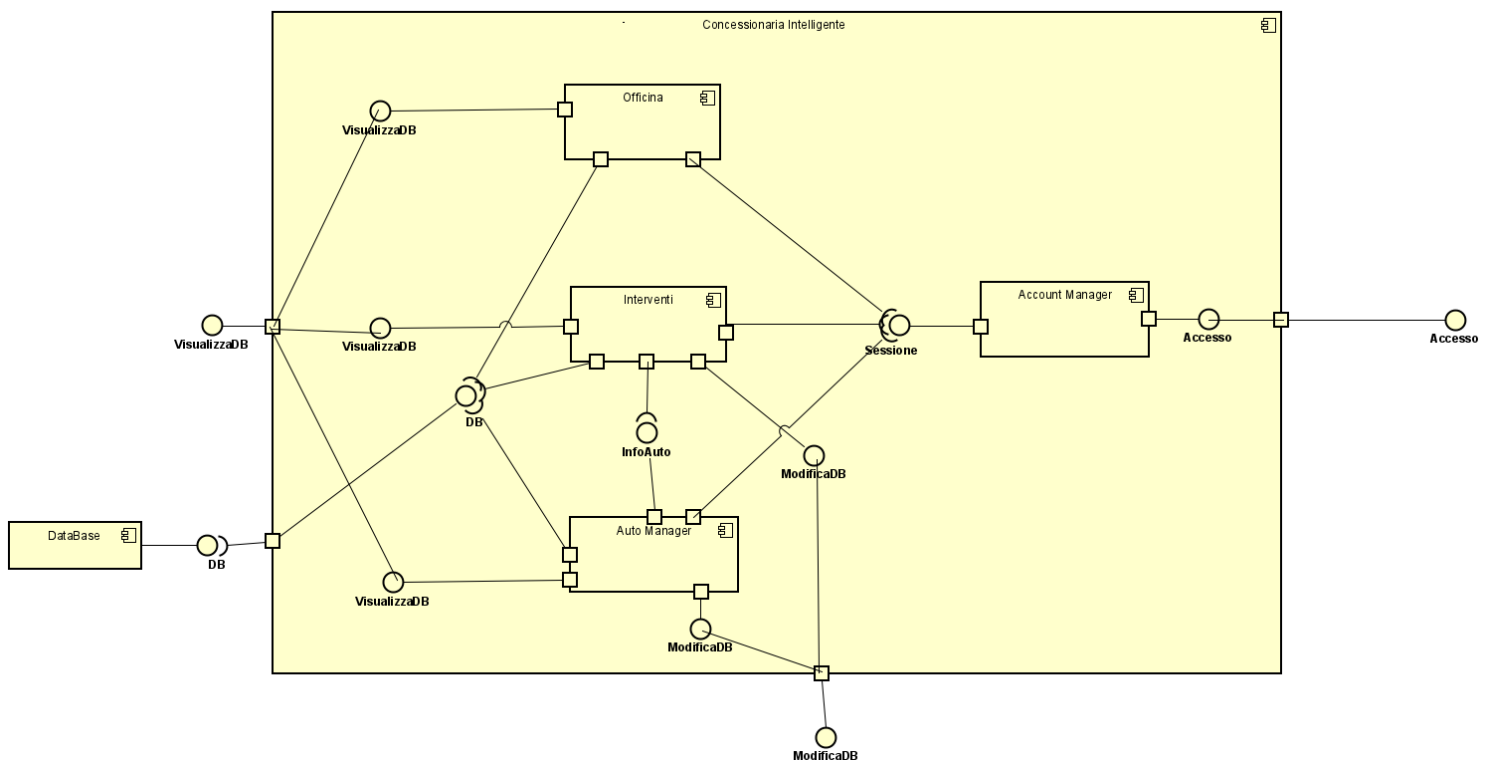


Figure 5: Iterazione 1 - Diagramma delle componenti

Il diagramma della *figura 6* è una vista più dettagliata di quello precedente; infatti, sono rappresentati anche i sottocomponenti delle parti principali.

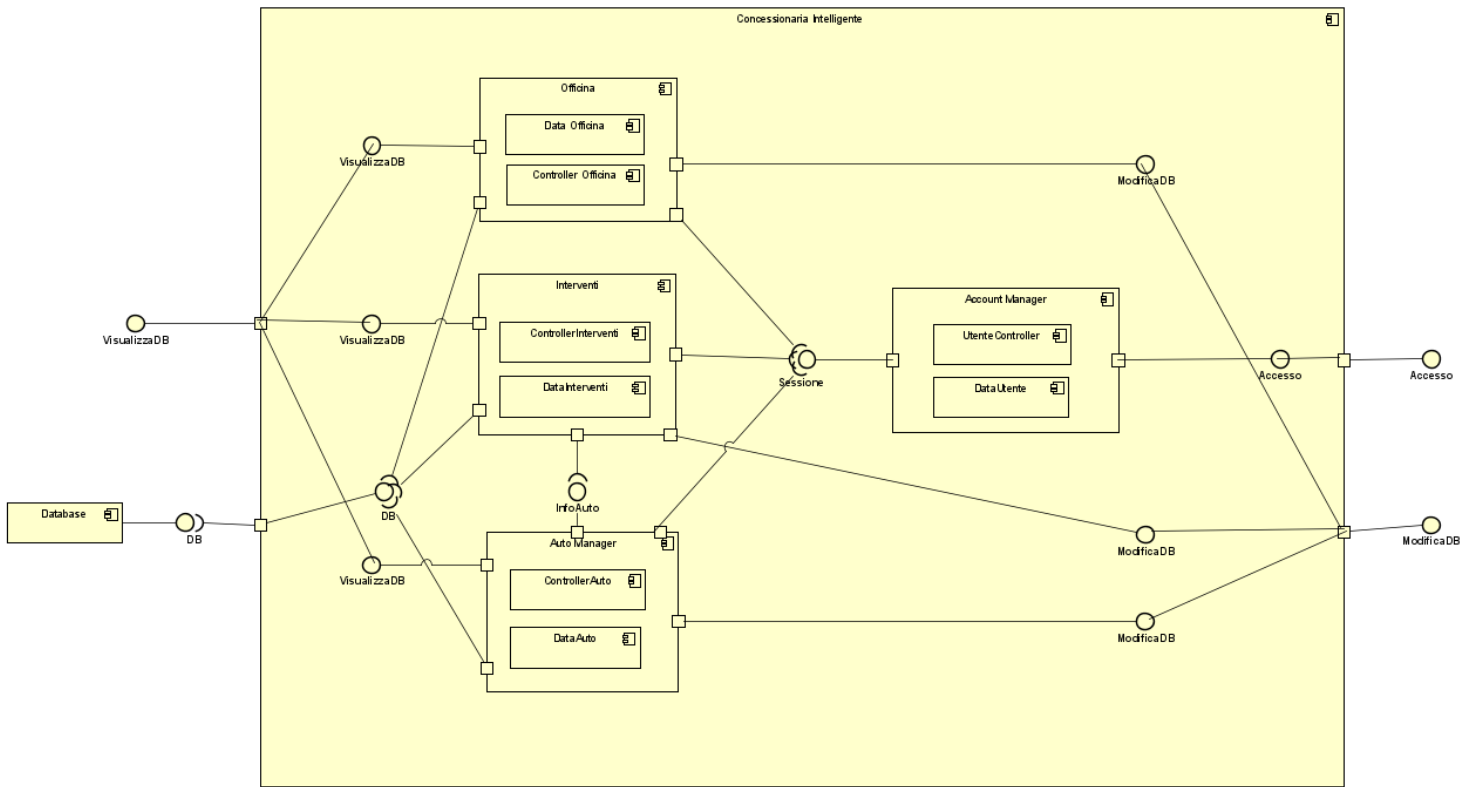


Figure 6: Iterazione 1 - Diagramma delle componenti completo

Diagramma UML delle classi per definire data type classes

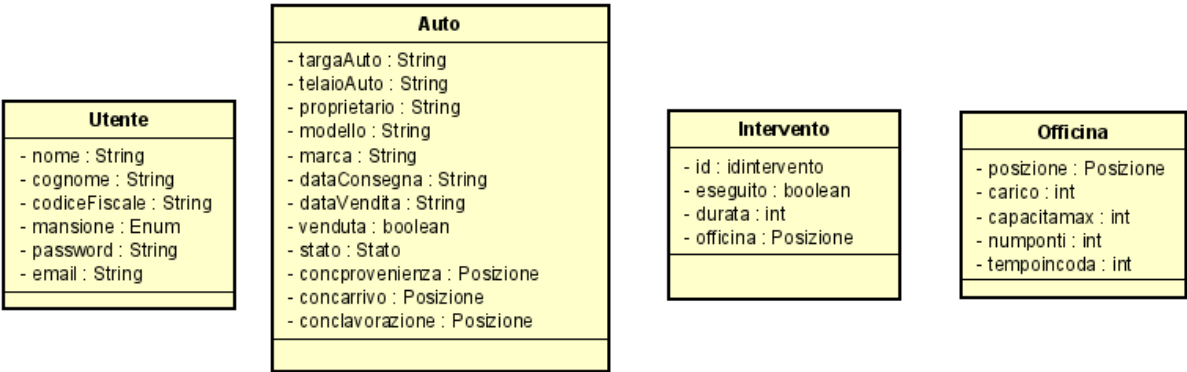


Figure 7: Iterazione 1 - Diagramma delle classi (data type classes)

## Diagramma UML delle classi per definire le interfacce

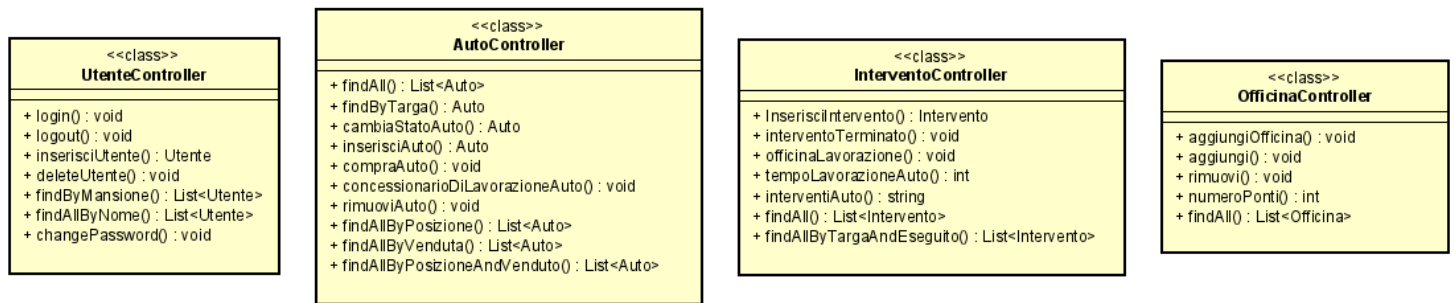


Figure 8 : Iterazione 1 - Diagramma delle classi (interfacce)

## Pseudocodice e analisi complessità tempo

La struttura su cui si basa lo pseudocodice è formata da:

1. Lista concatenata delle officine
2. Lista concatenata dei ponti per ogni officina
3. Lista concatenata delle macchine in coda al relativo ponte

Di seguito un esempio per chiarire la struttura su cui si basa lo pseudocodice.

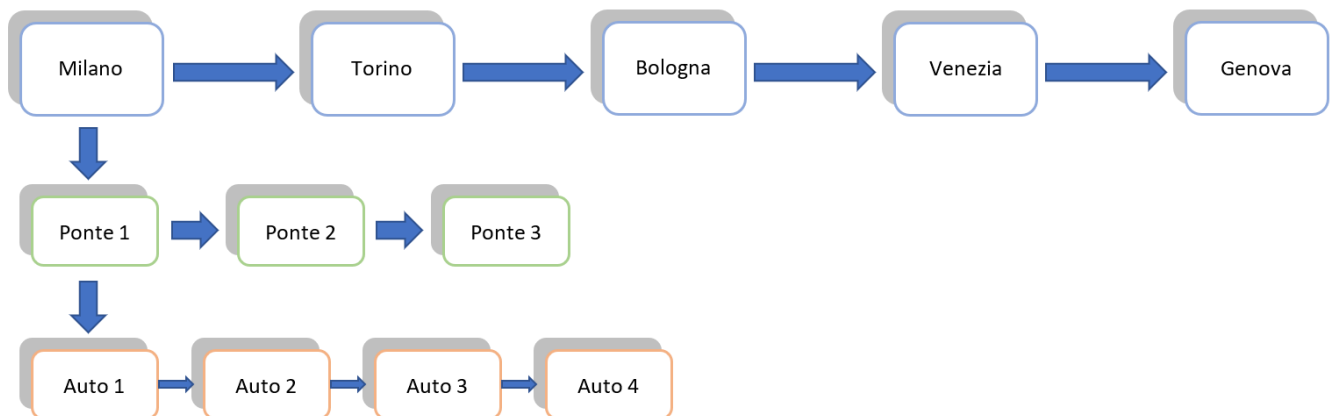


Figure 9: Iterazione 1 - Struttura pseudocodice

## Pseudocodice:

```
algoritmo schedulazione (lista auto, lista officine, lista interventi) → lista <targa, concessionario lavorazione>
ordina auto per data di consegna crescente
per ogni auto nella lista auto do
    per ogni officina della lista officine do
        if(carico dell'officina >= capacità massima dell'officina) then → continue
        per ogni ponte della lista ponti do
            per ogni slot del ponte do
                se trovo uno slot libero di durata maggiore o uguale al tempo di lavorazione dell'auto then
                    attribuisco lo slot alla macchina
```



Complessità temporale:

Se viene utilizzato un ordinamento delle macchine di tipo Merge sort, la complessità temporale è pari a  $O(m * \log(m))$ .

Se si hanno  $m$  macchine,  $n$  officine,  $p$  ponti in media e  $l$  macchine in coda per ogni ponte in media si ha:

$$T(m, n, p, l) = \sum_{i=0}^m \sum_{j=0}^n \sum_{k=0}^p \sum_{h=0}^l 1 = O(m * n * p * l)$$

Dato che il numero di macchine è molto più grande del numero di officine, di ponti e di auto in coda ad ogni ponte, si ottiene:  $T(m, n, p, l) = O(m)$ .

Questo risultato va sommato alla complessità temporale dell'ordinamento:

$$T(m) = O(m * \log(m)) + O(m) = O(m * \log(m))$$

## Analisi statica

Attraverso l'analisi statica si è voluto verificare che le varie componenti nella prima fase di sviluppo fossero totalmente indipendenti e raccogliessero tutte le classi necessarie alla struttura a microservizi.



Figure 10: Iterazione 1 - Analisi statica

## Analisi dinamica

Attraverso l'analisi dinamica si può controllare il grado di copertura delle istruzioni e attraverso i test si è in grado di verificare se il codice effettua le operazioni che ci si aspettava.

La copertura ottenuta non è molto alta (pari al 75,8 %), perché non è stata ancora implementata la parte relativa alle officine, come si può anche vedere dal numero di righe presenti nel package.















Element	Coverage	Covered Instru...	Missed Instruct...	Total Instructio...
▼  Conces_Int_v1.0	 75,8 %	1.131	361	1.492
▼  src/main/java	 69,1 %	796	356	1.152
>  com.autoManager	 72,5 %	327	124	451
>  com.accountManager	 77,7 %	272	78	350
>  com.interventi	 68,8 %	170	77	247
>  com.officina	 24,2 %	24	75	99
>  com	 60,0 %	3	2	5

Figure 11: Iterazione 1 - Analisi dinamica

## *Iterazione intermedia*

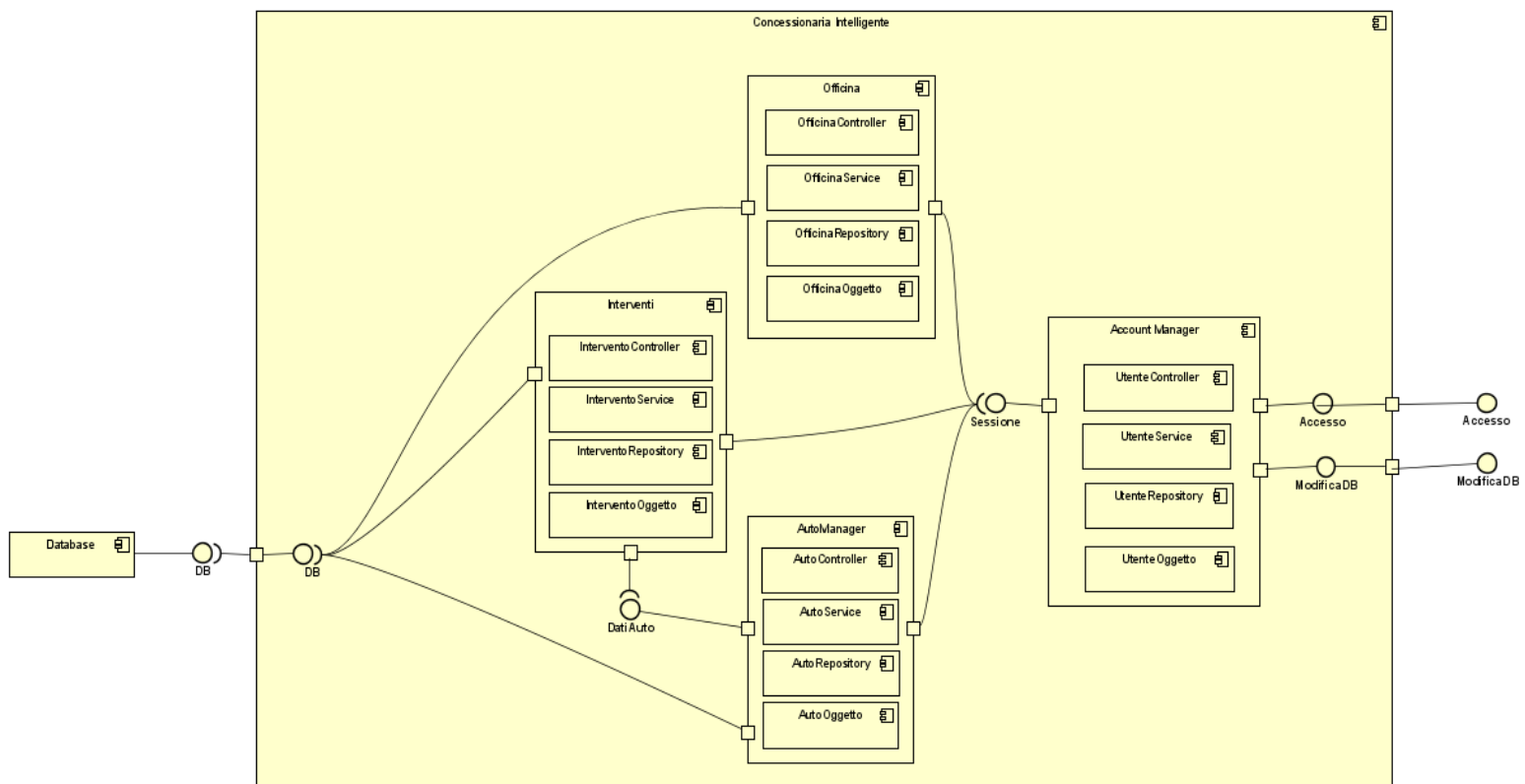
In questa iterazione è stato aggiunto l'algoritmo per schedare le macchine e l'implementazione delle componenti è stata estesa, in modo che nelle ultime iterazioni si dovranno aggiungere solo le interfacce con cui i diversi operatori interagiranno.

### Analisi requisiti e casi d'uso

Abbiamo mantenuto gli stessi casi d'uso delle iterazioni precedenti, perché già abbastanza completi e sufficienti.

### Design dell'architettura software

A questa iterazione, nel diagramma delle componenti, sono stati specificati i diversi elementi essenziale di ogni sottocomponente per far risaltare l'utilizzo della struttura a microservizi utilizzata.



*Figure 12: Iterazione intermedia - Diagramma delle componenti*

## Analisi statica

Rispetto all'analisi statica che è stata descritta all'iterazione precedente, le componenti comunicano tra di loro e si scambiano dati. In particolare, si può notare che *accountManager* interagisce di meno con le altre componenti perché non è stata implementata a sufficienza, in quanto si è deciso di non sviluppare la parte riguardante gli account e il login. *Algoritmo*, invece, comunica molto con le altre componenti, in quanto ha bisogno dei dati relativi alle officine, alle macchine e agli interventi per poter determinare la schedulazione delle auto.

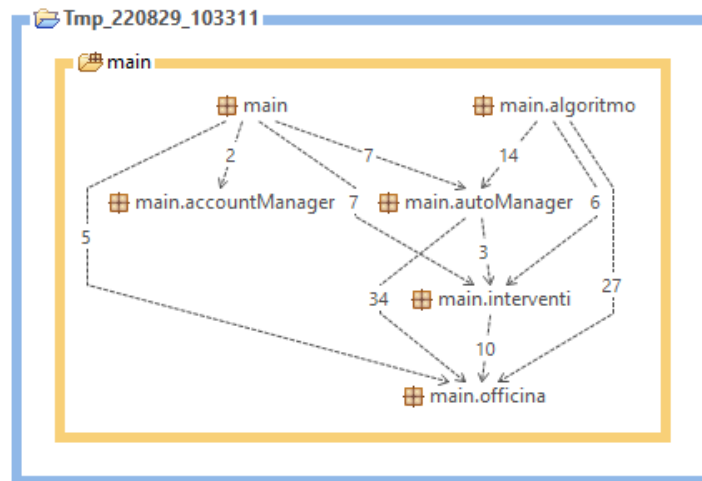


Figura 1: Iterazione intermedia - Analisi statica

## Analisi dinamica

Attraverso l'analisi dinamica si può controllare il grado di copertura delle istruzioni e attraverso i test si è in grado di verificare se il codice effettua le operazioni che ci si aspettava.

Abbiamo eseguito un test per ogni controller cercando di simulare il numero maggiore di situazioni possibili e abbiamo ottenuto una copertura dell'intero sistema pari a 75,1 %.

Element	Coverage	Covered Instru...	Missed Instruct...	Total Instructio...
Conces_Int_v2.0	84,0 %	4.464	850	5.314
src/main/java	75,1 %	2.534	841	3.375
main.algoritmo	85,6 %	1.484	249	1.733
main.autoManager	72,1 %	410	159	569
main.officina	54,8 %	165	136	301
main.interventi	60,4 %	200	131	331
main	3,3 %	3	88	91
main.accountManager	77,7 %	272	78	350

Figure 13: Iterazione intermedia - Analisi dinamica (Coverage)

## *Iterazione finale*

In questa iterazione sono state aggiunte le interfacce con cui gli utenti utilizzano il programma. In particolare, sono state aggiunte quelle di: meccanico, venditore e capo officina.

### Analisi requisiti e casi d'uso

Nel seguito verrà descritto nel dettaglio ogni caso d'uso implementato con le interfacce. I codici utilizzati per identificarli sono composti dalla sigla dell'utente che può eseguirli e viene aggiunto un numero che segue la numerazione usata in precedenza. Ad esempio: inserimento nuovo intervento avrà codice PV1a che sta a significare che l'attore che può eseguire queste azioni è PV (post vendita), il numero 1 indica che è nel primo blocco di azioni e invece l'ultima lettera rappresenta la sotto categoria del blocco.

ID:	V1
Nome:	Visualizzare database contenente le macchine in vendita
Attore/i:	Venditore
Triggered by:	Il venditore preme l'apposito pulsante aggiorna per visualizzare le macchine in vendita
Precondizione:	Login
Sequenza di azioni:	<ol style="list-style-type: none"><li>1. Il venditore deve effettuare il login con le sue credenziali</li><li>2. Sceglie la filiale in cui si trova</li><li>3. Preme il pulsante aggiorna</li></ol>
Postcondizione:	Logout

ID:	V2
Nome:	Vendere una macchina
Attore/i:	Venditore
Triggered by:	Il venditore preme l'apposito pulsante vendita per poter effettuare la vendita della macchina
Precondizione:	Login
Sequenza di azioni:	<ol style="list-style-type: none"><li>1. Il venditore deve effettuare il login con le sue credenziali</li><li>2. Sceglie la filiale in cui si trova</li><li>3. Inserisce i dati della macchina: targa, telaio, data consegna, data vendita</li><li>4. Preme il pulsante vendita</li></ol>

Postcondizione:	Logout
Eccezione/i:	3.a se vengono inseriti dei dati sbagliati viene sollevata eccezione perché l'auto non esiste 3.b se vengono inseriti dati di un'auto già venduta, viene sollevata un'eccezione

ID:	CO1
Nome:	Visualizza la schedulazione della sua filiale
Attore/i:	Capo officina
Triggered by:	Il capo officina preme l'apposito pulsante aggiorna per poter visualizzare gli interventi
Precondizione:	Login
Sequenza di azioni:	<ol style="list-style-type: none"> <li>1. Il capo officina deve effettuare il login con le sue credenziali</li> <li>2. Sceglie la posizione della filiale</li> </ol>
Postcondizione:	Logout

ID:	CO2
Nome:	Cambia stato delle auto
Attore/i:	Capo officina
Triggered by:	Il capo officina preme l'apposito pulsante per poter salvare la modifica del cambio stato della macchina inserita
Precondizione:	Login
Sequenza di azioni:	<ol style="list-style-type: none"> <li>1. Il capo officina deve effettuare il login con le sue credenziali</li> <li>2. Scegliere la posizione della filiale</li> <li>3. Inserire la targa della macchina e scegliere lo stato che deve assumere la macchina</li> <li>4. Premere il pulsante cambia</li> </ol>
Postcondizione:	Logout
Eccezione/i:	3.a viene inserita una targa non associata ad una macchina nel database

ID:	M1
Nome:	Visualizza le macchine da sistemare (relative al suo ponte)
Attore/i:	Meccanico
Triggered by:	Il meccanico preme l'apposito pulsante visualizza per poter vedere le macchine da sistemare relative al suo ponte
Precondizione:	Login
Sequenza di azioni:	<ol style="list-style-type: none"> <li>1. Il meccanico deve effettuare il login con le sue credenziali</li> <li>2. Scegliere la posizione della filiale</li> <li>3. Scegliere il ponte che gli è stato assegnato</li> <li>4. Premere il pulsante per visualizzare le macchine</li> </ol>
Postcondizione:	Logout
Eccezione/i:	3.a non si sceglie nessun ponte

ID:	M2
Nome:	Visualizza gli interventi da effettuare
Attore/i:	Meccanico
Triggered by:	Il meccanico preme l'apposito pulsante visualizza per poter vedere gli interventi che deve effettuare sulla prima macchina dell'elenco
Precondizione:	Login
Sequenza di azioni:	<ol style="list-style-type: none"> <li>1. Il meccanico deve effettuare il login con le sue credenziali</li> <li>2. Scegliere la posizione della filiale</li> <li>3. Scegliere il ponte che gli è stato assegnato</li> <li>4. Premere il pulsante per visualizzare gli interventi che deve effettuare sulla prima macchina dell'elenco</li> </ol>
Postcondizione:	Logout
Eccezione/i:	

ID:	M3
Nome:	Segnalazione fine intervento
Attore/i:	Meccanico
Triggered by:	Il meccanico preme l'apposito pulsante effettuato per segnalare la fine degli interventi relativi a una macchina
Precondizione:	Login





## Analisi dinamica

Attraverso l'analisi dinamica si può controllare il grado di copertura delle istruzioni e attraverso i test si è in grado di verificare se il codice effettua le operazioni che ci si aspettava.

Abbiamo eseguito un test per ogni controller cercando di simulare il numero maggiore di situazioni possibili.

Non è stato possibile effettuare il test del main, perché contiene le interfacce GUI. Per questo motivo se ricalcoliamo la coverage, escludendo il package main, otteniamo: 75,21 % di coverage.

Element	Coverage	Covered Instru...	Missed Instruct...	Total Instructio...
Conces_Int_v2.0	69,3 %	5.557	2.458	8.015
src/main/java	56,1 %	3.131	2.452	5.583
> main	0,6 %	8	1.423	1.431
> main.algoritmo	77,4 %	1.689	493	2.182
> main.autoManager	70,0 %	573	245	818
> main.interventi	58,4 %	260	185	445
> main.accountManager	77,7 %	272	78	350
> main.officina	92,2 %	329	28	357

Figure 15: Iterazione finale - Analisi dinamica (coverage)

Abbiamo, anche, eseguito una verifica manuale dell'algoritmo con gli stessi dati usati nel test.

Nella prima tabella sottostante sono riportati i dati utilizzati per testare l'algoritmo; per ogni targa sono riportati anche la durata totale degli interventi, l'officina di partenza e di arrivo. Uno dei requisiti era che una macchina venisse schedulata in una sola officina per tutti gli interventi; si è considerata, quindi, solo la somma complessiva dei tempi degli interventi come un unico grande intervento da effettuare. Un'altra assunzione è quella di non considerare le pause dei meccanici o le giornate lavorative, ma ogni volta l'algoritmo restituisce un elenco di lavori da effettuare in ordine di priorità, ma senza date effettive.

Targa	Durata intervento	Officina di partenza	Officina di arrivo
targa00	240	Torino	Bologna
targa01	150	Milano	Torino
targa02	120	Venezia	Milano
targa03	110	Venezia	Torino
targa04	60	Milano	Milano
targa05	40	Genova	Bologna
targa06	180	Milano	Bologna
targa07	150	Torino	Genova
targa08	200	Bologna	Genova
targa09	160	Torino	Milano
targa10	240	Milano	Milano
targa11	60	Milano	Torino
targa12	60	Bologna	Genova
targa13	60	Venezia	Venezia

Targa	Durata intervento	Officina di partenza	Officina di arrivo
targa14	120	Milano	Milano
targa15	90	Milano	Torino
targa16	90	Milano	Torino
targa17	600	Torino	Torino
targa18	30	Genova	Torino
targa19	60	Bologna	Milano
targa20	10	Milano	Torino
targa21	60	Bologna	Torino
targa22	10	Bologna	Torino
targa23	80	Bologna	Bologna
targa24	150	Bologna	Bologna
targa25	220	Bologna	Bologna
targa26	100	Genova	Bologna
targa27	30	Genova	Torino

Per poter eseguire la schedulazione è necessario conoscere le capacità delle singole città, il numero di ponti e anche la distanza tra ogni città.

<b>Città</b>	<b>Capacità massima</b>	<b>Numero ponti</b>
Milano	13	4
Torino	10	4
Bologna	10	3
Venezia	2	1
Genova	3	2

	Milano	Torino	Bologna	Venezia	Genova
Milano	0	143	216	270	150
Torino	143	0	332	403	171
Bologna	216	332	0	154	296
Venezia	270	403	154	0	401
Genova	150	171	296	401	0

Con tutte queste informazioni è stato possibile determinare la schedulazione nelle varie città come segue.

<b>Milano</b>							
Ponte 1		Ponte 2		Ponte 3		Ponte 4	
targa01	150	targa04	60	targa06	180	targa10	240
libero	120	targa20	10	targa14	120		
targa02	120	libero	73				
		targa09	160				

<b>Torino</b>							
Ponte 1		Ponte 2		Ponte 3		Ponte 4	
targa00	240	targa07	150	libero	143	libero	143
libero	92	targa16	90	targa11	60	targa15	90
targa21	60	libero	92	targa17	600		
libero	11	targa22	10				
targa03	110						

<b>Bologna</b>					
Ponte 1		Ponte 2		Ponte 3	
targa08	200	targa12	60	targa19	60
libero	96	targa23	80	targa24	150
targa05	40	targa25	220	libero	86
				targa26	100

<b>Venezia</b>	
Ponte 1	
targa13	60

<b>Genova</b>			
Ponte 1		Ponte 2	
targa18	30	targa27	30

Si può osservare che tutte le capacità delle officine sono state rispettate e ogni macchina è stata schedulata in modo che venisse riparata nel più breve tempo possibile, anche se ciò comporta dover spostarla in un'altra officina.

# Guida all'installazione dell'applicazione

Per installare il sistema serve scaricare lo zip contenente l'intero codice dell'ultima iterazione e il file contenente il database. Noi abbiamo usato i seguenti tool: eclipse e XAMPP; quest'ultimo è una multiplatforma software che permette di avere sia il web server Apache sia il database server MySQL.

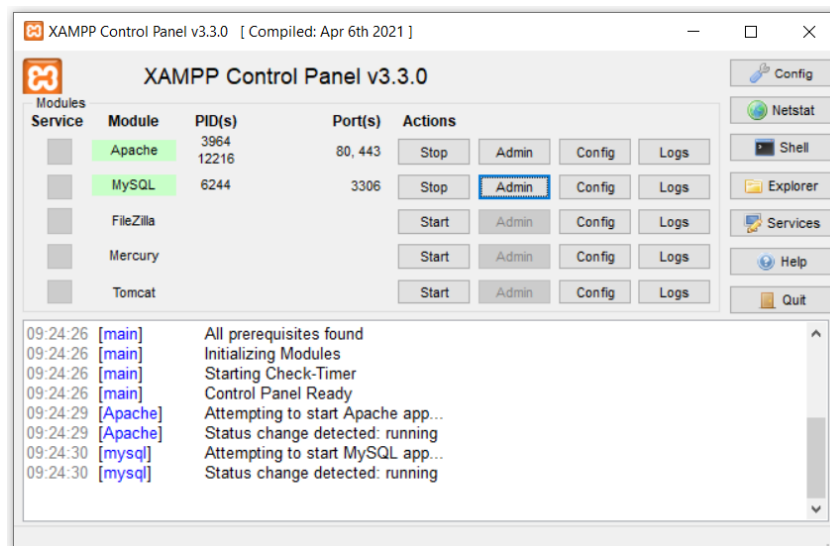


Figure 16: Multiplatforma software XAMPP

Una volta avviati sia Apache che MySQL, bisogna premere il pulsante Admin di MySQL per accedere a phpMyAdmin (applicazione web che consente di amministrare il database tramite un browser). A questo punto è necessario creare un database nominato 'concessionario\_intelligente' e importare l'intero database al suo interno.

L'applicazione ora può essere eseguita. Attenzione che una volta aperta l'applicazione non deve essere chiusa, perché l'algoritmo non è ancora in grado di salvare l'intera schedulazione per le esecuzioni successive.

Il database è composto da quattro tabelle:

Tabella	Azione	Righe	Tipo
<input type="checkbox"/> auto	Mostra Struttura Cerca Inserisci Svuota Elimina	20	InnoDB
<input type="checkbox"/> intervento	Mostra Struttura Cerca Inserisci Svuota Elimina	32	InnoDB
<input type="checkbox"/> officina	Mostra Struttura Cerca Inserisci Svuota Elimina	5	InnoDB
<input type="checkbox"/> utente	Mostra Struttura Cerca Inserisci Svuota Elimina	0	InnoDB

Figure 17: Tabelle che costituiscono il database Concessionario\_intelligente

La tabella *utente* non contiene righe perché si è deciso di non implementare la parte relativa agli accessi.

# Sviluppi futuri

Alcuni possibili sviluppi futuri:

1. Implementare il componente *Account Manager* in modo da gestire gli accessi e gli utenti; grazie a questo si possono registrare le variazioni nel database con il relativo utente che le ha effettuate.
2. Implementare i casi d'uso mancanti in modo che tutte le tipologie di utenti possano accedere al sistema.
3. Sviluppare un proprio database con un'interfaccia apposita e che permetta anche di inserire ulteriori informazioni come, ad esempio, le foto della macchina e il libretto dell'auto.
4. Salvare la schedulazione completa in modo che l'applicazione possa essere chiusa e una volta aperta è possibile rivedere la schedulazione delle officine e dei ponti anche se non sono state aggiunte macchine o lavorazione e cioè non è avvenuta una nuova schedulazione.
5. Aggiungere nell'interfaccia del capo filiale a quale meccanico attribuire ogni ponte, in modo che ciascun meccanico quando effettua il login non deve scegliere il ponte, ma vede direttamente le macchine che deve sistemare e i lavori della prima macchina nell'elenco.
6. Al posto delle interfacce utente si potrebbero esporre solo i servizi in modo da poter avere le interfacce personalizzate a seconda delle necessità.

# Uso dell'applicazione

La prima schermata che appare eseguendo il programma è il menu principale in cui si può scegliere la posizione in cui ci si trova (Milano, Torino, Bologna, Genova, Venezia) ed effettuare il login a seconda della mansione.

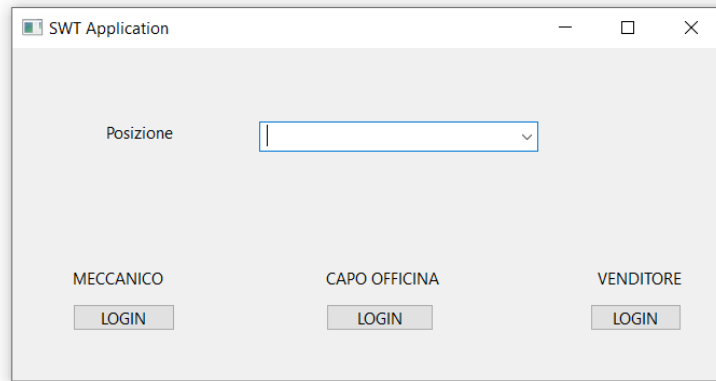


Figura 2: Interfaccia menu principale

Non avendo implementato il login degli utenti, quando si schiaccia il bottone ‘login’ si entra direttamente nell’interfaccia relativa alla mansione specificata sopra senza dover inserire le credenziali e la password.

Tutte le interfacce relative alla singola mansione hanno il pulsante ‘logout’ per poter uscire e tornare al menu principale.

## Interfaccia meccanico

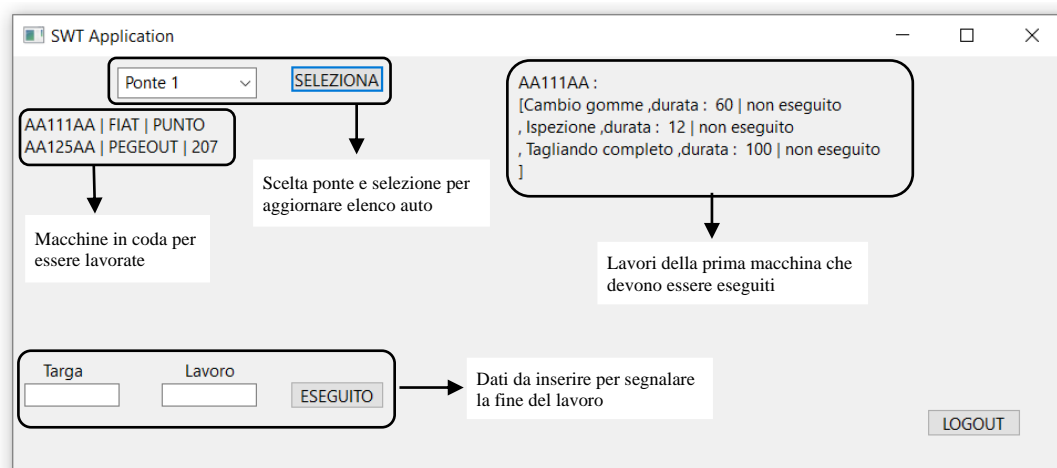


Figura 3: Interfaccia meccanico (Officina Milano, ponte 1)

Il meccanico deve scegliere il ponte che gli è stato assegnato dal capo officina dall’apposita combo box, in questo modo schiacciando il pulsante ‘seleziona’ vengono visualizzate le macchine che

devono essere lavorate. Per ogni automobile vengono visualizzate le seguenti informazioni: targa, marca e modello.

Nella parte destra dell'interfaccia compaiono i lavori relativi alla prima macchina da aggiustare che deve eseguire il meccanico; in particolare vengono specificati la targa, il lavoro da fare, la durata stimata e se è già stato eseguito o meno.

In fondo alla schermata sono presenti due box di testo che servono per inserire la targa e il lavoro che è appena stato eseguito per poterlo segnare come terminato attraverso il pulsante 'eseguito'.

## Interfaccia venditore

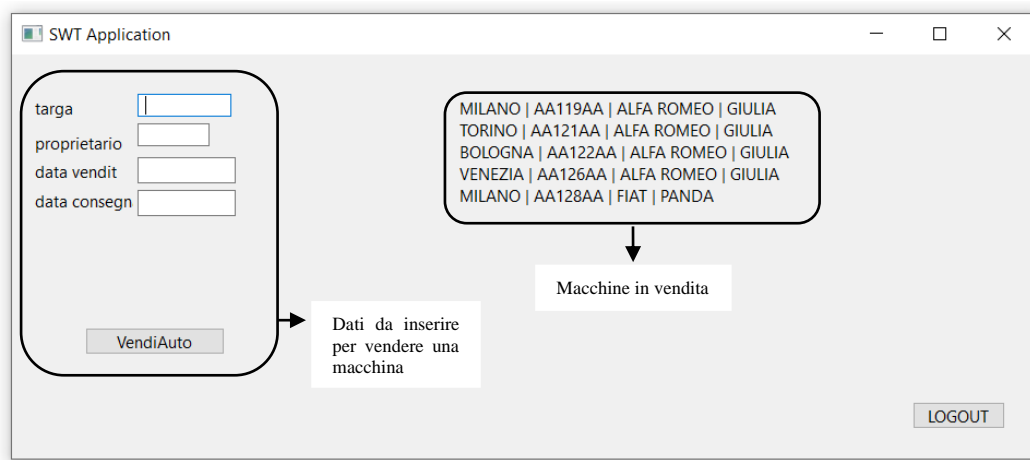


Figura 4: Interfaccia venditore a Milano

Nella parte destra sono presenti tutte le macchine in vendita con i relativi dati: concessionaria in cui si trova attualmente, targa, marca, modello. Nella prima parte, invece, sono presenti le box di testo per inserire i dati relativi per poter segnalare la vendita della macchina.

## Interfaccia capo officina

La prima schermata che appare contiene solo i titoli dei contenuti che verranno mostrati una volta che si preme sul pulsante 'AGGIORNA'.

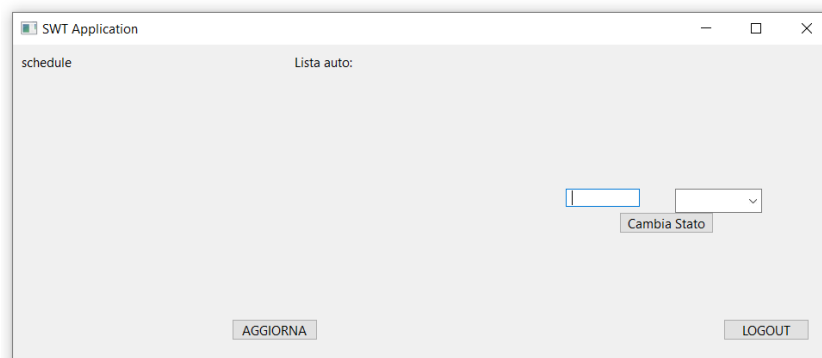


Figure 18: Interfaccia iniziale capo officina di Milano

Ecco come appare l'interfaccia dopo che è stato lanciato l'algoritmo di schedulazione:

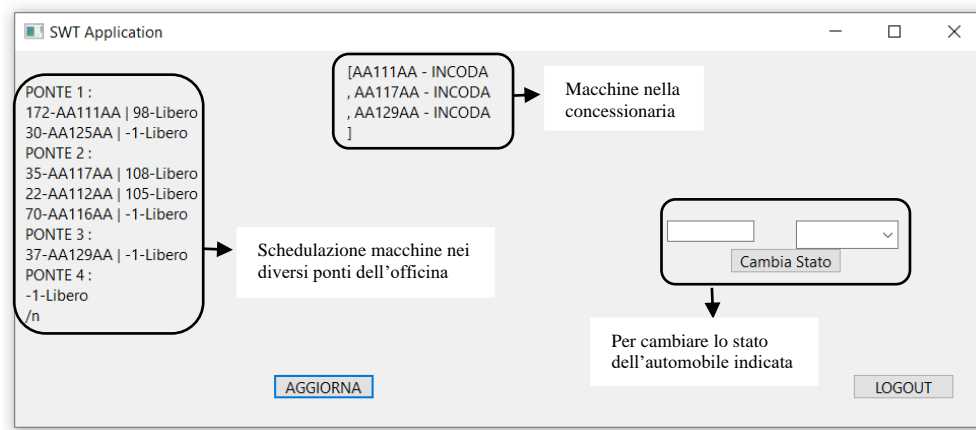


Figura 5: Interfaccia capo officina concessionaria Milano

Il capo officina, nella parte sinistra della schermata, visualizza la schedulazione delle macchine nei diversi ponti per la sua officina.

Nella parte centrale, invece, visualizza le macchine presenti nella concessionaria e che sono in coda per essere lavorate o devono essere trasportate in un'altra officina per essere lavorate in meno tempo.

Infine, a destra ci sono due box per cambiare lo stato di un'automobile, in particolare nella box di testo si inserisce la targa e nella combo box si seleziona il nuovo stato; nel momento in cui si preme sul pulsante 'cambia stato', viene aggiornato il database. Questo vuol dire che se una macchina era presente in una determinata officina e si cambiasse lo stato in trasporto, questa auto verrebbe visualizzata nell'interfaccia del capo officina della concessionaria in cui deve arrivare.