***Progettazione e algoritmi***

***Progetto: concessionario intelligente***



Maffi Andrea,

Sassi Annamaria, 1063453

Testa Lorenzo,

Anno accademico 2021-2022

Sommario

[Concessionario intelligente 1](#_Toc112753591)

[Iterazione 0 2](#_Toc112753592)

[Analisi dei requisiti 2](#_Toc112753593)

[Configurazione iniziale architettura 2](#_Toc112753594)

[Modellazione casi d’uso 3](#_Toc112753595)

[Tool chain 5](#_Toc112753596)

[Iterazione 1 6](#_Toc112753597)

[Casi d’uso 6](#_Toc112753598)

[Diagramma UML di package 12](#_Toc112753599)

[Diagramma UML delle componenti 13](#_Toc112753600)

[Diagramma UML delle classi per definire data type classes 15](#_Toc112753601)

[Diagramma UML delle classi per definire le interfacce 15](#_Toc112753602)

[Pseudocodice e analisi complessità tempo 15](#_Toc112753603)

[Analisi statica 16](#_Toc112753604)

[Iterazione intermedia 17](#_Toc112753605)

[Analisi requisiti e casi d’uso 17](#_Toc112753606)

[Design dell’architettura software 17](#_Toc112753607)

[Analisi statica 17](#_Toc112753608)

[Analisi dinamica 18](#_Toc112753609)

[Iterazione finale 19](#_Toc112753610)

[Analisi requisiti e casi d’uso 19](#_Toc112753611)

[Design dell’architettura software 19](#_Toc112753612)

[Analisi statica 19](#_Toc112753613)

[Analisi dinamica 19](#_Toc112753614)

[Guida all’installazione dell’applicazione 20](#_Toc112753615)

[Uso dell’applicazione 21](#_Toc112753616)

# 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 (può aumentare o diminuire il carico), cancellati oppure aggiunti; in tutti e tre i casi viene aggiornato lo slot di lavorazione.

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.

Nella filiale centrale è presente il capo di tutte le filiali e può controllare le macchine in transito tra le filiali, quelle in lavorazione, quelle in attesa di essere trasportate.

Per poter accedere all’applicazione web del sistema bisogna accedere con le proprie credenziali.

L’amministratore di rete può inserire nuovi utenti, eliminare account o modificare i dati di un utente.

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.

## Iterazione 0

### Analisi dei requisiti

Il sistema è composto da: una serie di interfacce con cui le persone possono interagire, le varie componenti sottostanti che permettono ciò e l’algoritmo. Grazie a quest’ultimo è possibile determinare la schedulazione delle macchine vendute in modo che siano aggiustate nel più breve tempo possibile.

### Configurazione iniziale architettura

Abbiamo pensato a un’architettura nella quale ogni sottosistema rappresenta un tipo di utente e queste componenti interagiscono con il database della singola filiale.

Dato che ci sono più filiali, abbiamo individuato una filiale centrale nella quale c’è il server centrale e il capo filiale centrale che può controllare le altre filiali.

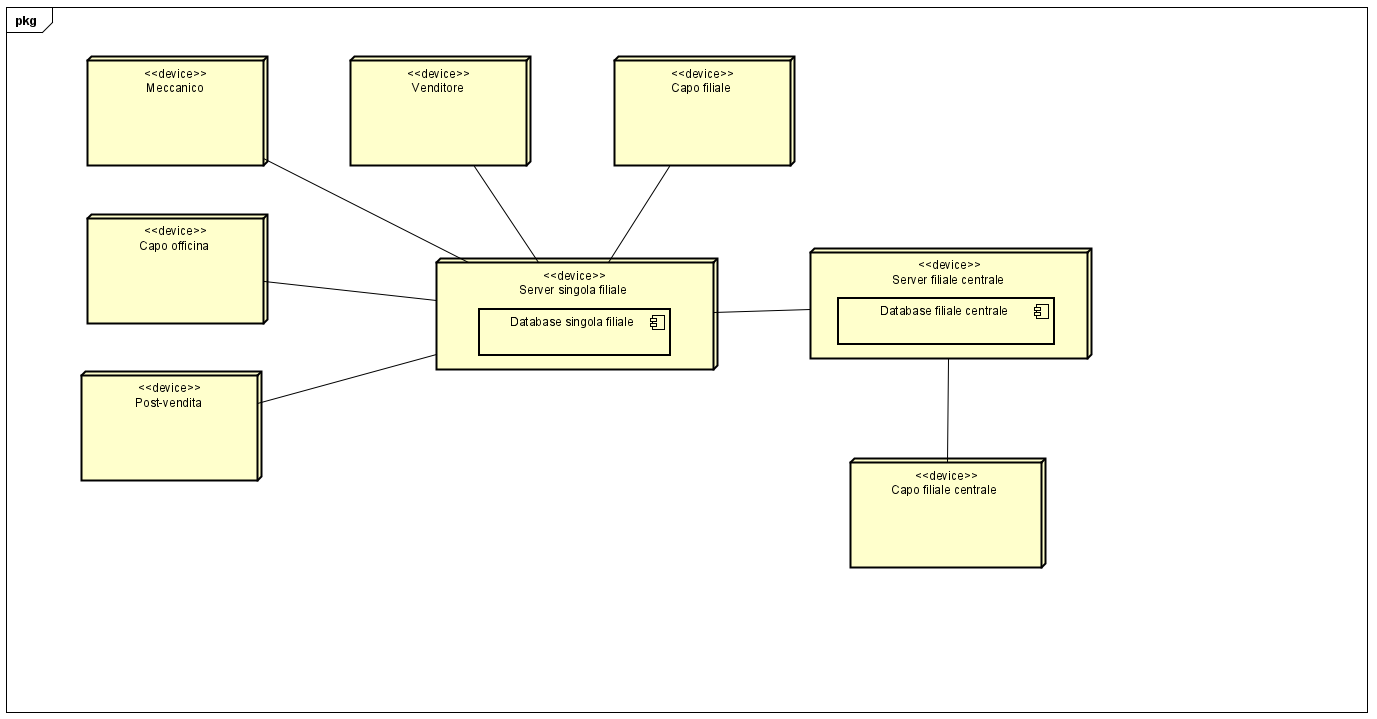


Figura : Architettura di base

### 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.

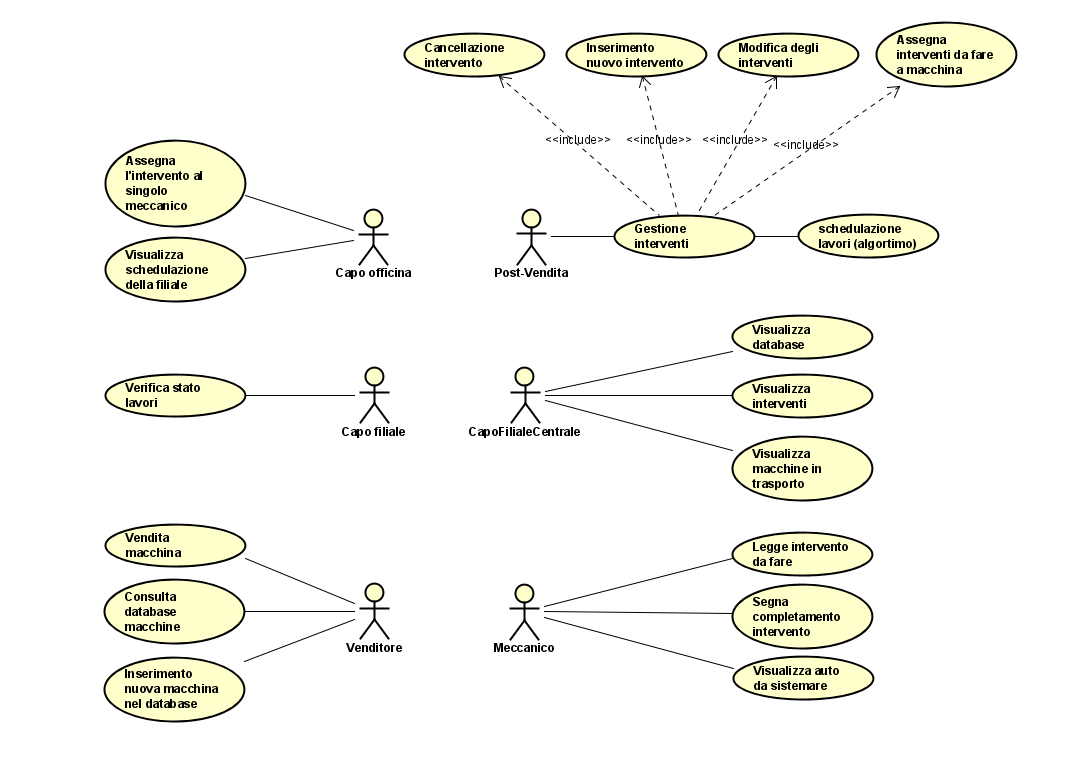


Figura : Iterazione 0 - casi d'uso

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) |

|  |
| --- |
| Nome: V |
| Attori: Venditore |
| Eventi:   1. Visualizzare database contenente le 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. Cambia 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 gli interventi programmati 2. Visualizza le macchine vendute 3. Visualizza le macchine in vendita |

|  |
| --- |
| 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 3. Visualizza le macchine in vendita |

### Tool chain

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

## 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 che gestisce gli account, infatti ogni utente per poter accedere al programma deve per prima cosa effettuare il login. Nel momento in cui si accede attraverso il login, il sistema conosce la mansione ricoperta dall’individuo e quindi gli permette di effettuare solo le azioni associate alla sua mansione. Di conseguenza, anche le interfacce che verranno visualizzate dipenderanno dal tipo di mansione ricoperta.

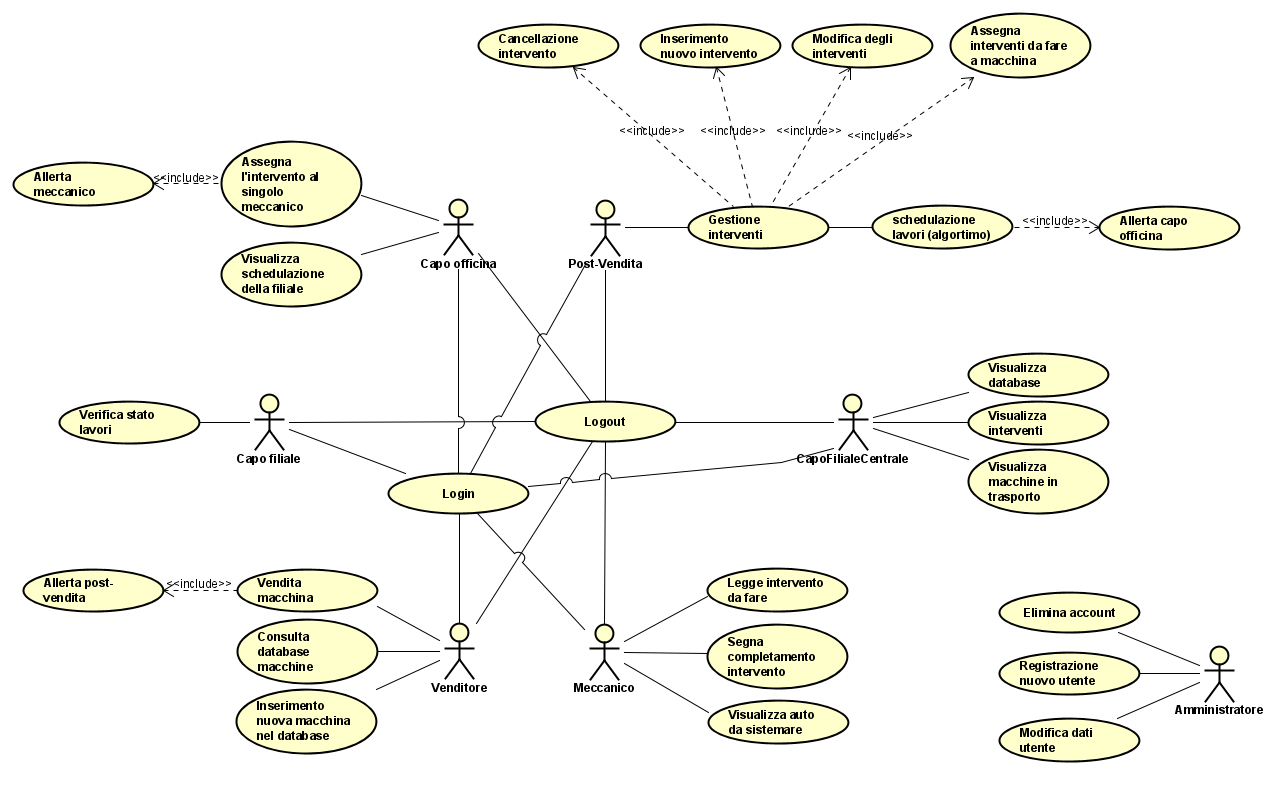


Figura : Iterazione 1 - casi d'uso

|  |
| --- |
| Nome: PV |
| Attori: post-vendita |
| Precondizioni: login |
| Eventi:   1. Gestione degli interventi    1. Cancellazione intervento    2. Inserimento nuovo intervento    3. Modifica degli interventi    4. Assegnare intervento a una macchina |
| Postcondizioni: logout |

|  |
| --- |
| Nome: V |
| Attori: Venditore |
| Precondizioni: login |
| Eventi:   1. Visualizzare database contenente le 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 gli interventi programmati 2. Visualizza le macchine vendute 3. Visualizza le macchine in vendita |
| 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 3. Visualizza le macchine in vendita |
| 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.

|  |
| --- |
| Interventi:   1. Gestione interventi    1. Assegnamento interventi alla singola macchina    2. Modifica di un intervento    3. Inserimento di un intervento    4. Cancellazione di un intervento 2. Determinazione slot intervento 3. Visualizzazione degli interventi programmati in una determinata filiale 4. Assegnazione interventi ai meccanici 5. Segnalazione del completamento dell’intervento |
| Automobile:   1. Visualizzare database contenente le macchine in vendita 2. Vendere una macchina 3. Inserimento nuova macchina in vendita nel database 4. Visualizzazione macchine vendute 5. Visualizzazione macchine in trasporto |
| Account Manager:   1. Login 2. Logout 3. Registrazione nuovo utente 4. Eliminazione account 5. Modifica dati utente |
| Officina:   1. Aggiungere un’officina 2. Eliminare un’officina 3. Aggiungere un ponte |

Nel seguito verrà descritto nel dettaglio ogni caso d’uso implementato. 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: | 1. Il venditore deve effettuare il login con le sue credenziali 2. Sceglie la filiale in cui si trova 3. Preme il pulsante aggiorna |
| Postcondizione: | Logout |
| Eccezione/i: |  |

|  |  |
| --- | --- |
| 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: | 1. Il venditore deve effettuare il login con le sue credenziali 2. Sceglie la filiale in cui si trova 3. Inserisce i dati della macchina: targa, telaio, data consegna, data vendita 4. Preme il pulsante vendita |
| 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: | 1. Il capo officina deve effettuare il login con le sue credenziali 2. Sceglie la posizione della filiale |
| Postcondizione: | Logout |
| Eccezione/i: |  |

|  |  |
| --- | --- |
| 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: | 1. Il capo officina deve effettuare il login con le sue credenziali 2. Scegliere la posizione della filiale 3. Inserire la targa della macchina e scegliere lo stato che deve assumere la macchina 4. Premere il pulsante cambia |
| 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: | 1. Il meccanico deve effettuare il login con le sue credenziali 2. Scegliere la posizione della filiale 3. Scegliere il ponte che gli è stato assegnato 4. Premere il pulsante per visualizzare le macchine |
| Postcondizione: | Logout |
| Eccezione/i: |  |

|  |  |
| --- | --- |
| 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: | 1. Il meccanico deve effettuare il login con le sue credenziali 2. Scegliere la posizione della filiale 3. Scegliere il ponte che gli è stato assegnato 4. Premere il pulsante per visualizzare gli interventi che deve effettuare sulla prima macchina dell’elenco |
| 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 |
| Sequenza di azioni: | 1. Il meccanico deve effettuare il login con le sue credenziali 2. Scegliere la posizione della filiale 3. Scegliere il ponte che gli è stato assegnato 4. Inserisce targa e intervento 5. Preme il bottone fine intervento |
| Postcondizione: | Logout |
| Eccezione/i: |  |

### Diagramma UML di package

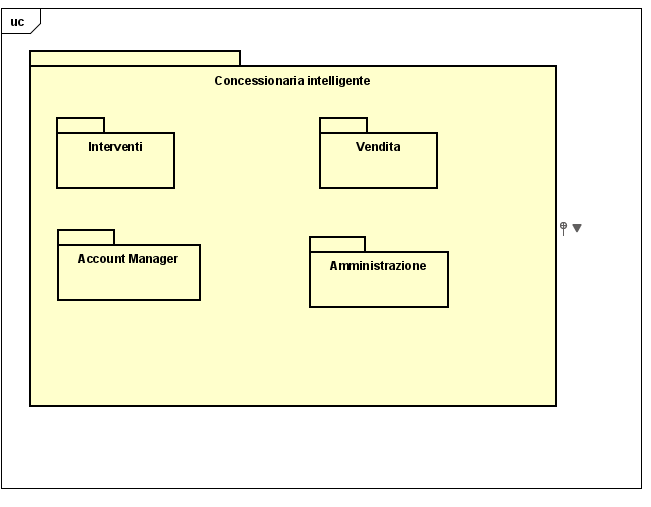


Figura : Iterazione 1 - Diagramma UML di package

### Diagramma UML delle componenti

I casi d’uso sono raggruppati in quattro macro-gruppi, per questo motivo abbiamo sviluppato quattro package indipendenti tra di loro e con ruoli diversi, in modo che siano coesi al loro interno, ma poco accoppiati tra di loro.

Ogni gruppo al suo interno segue il pattern esagonale sviluppato attraverso REST; quest’ultimo prevede una suddivisione in: controller, service ed entity. Il controller è all’esterno e serve per poter comunicare con l’esterno e nasconde al suo interno le altre componenti. I servizi, invece, servono per far comunicare le entità con i controller. Entity è l’oggetto che viene passato.

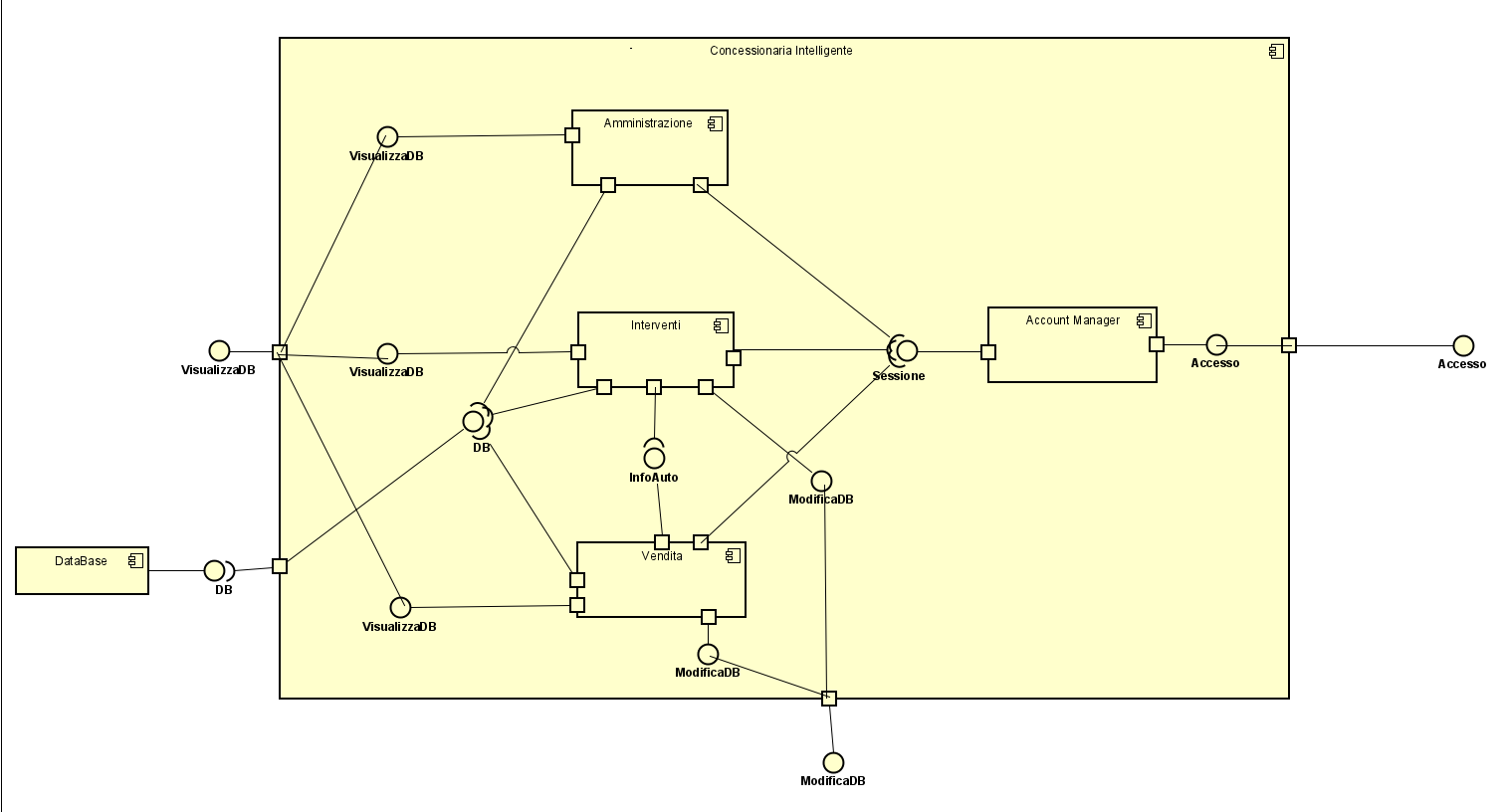


Figura : Iterazione 1 - Diagramma delle componenti

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

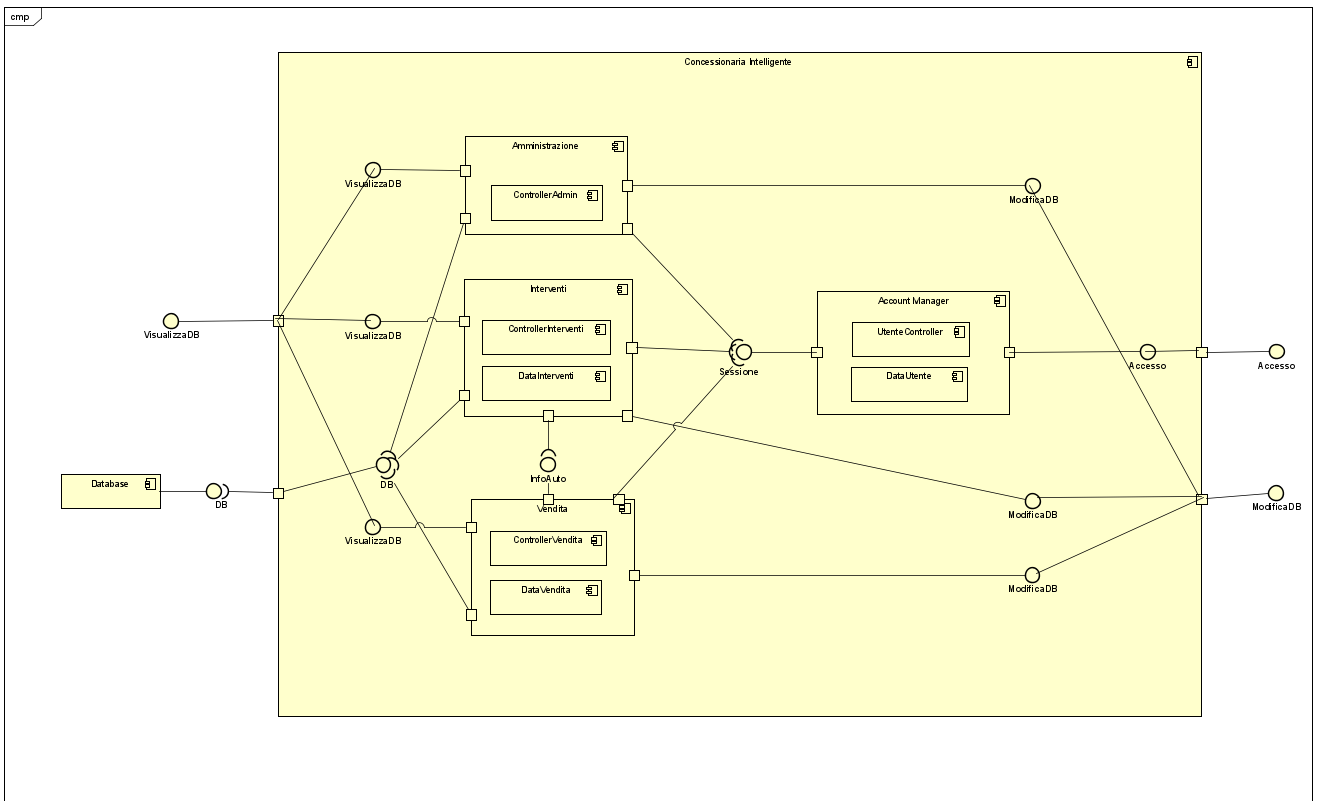


Figura : Iterazione 1 - Diagramma delle componenti completo

### Diagramma UML delle classi per definire data type classes

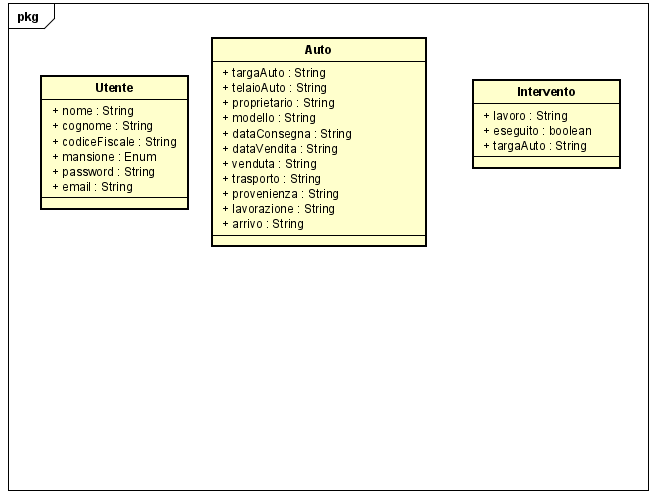


Figura : Iterazione 1 - Diagramma delle classi (data type classes)

### Diagramma UML delle classi per definire le interfacce

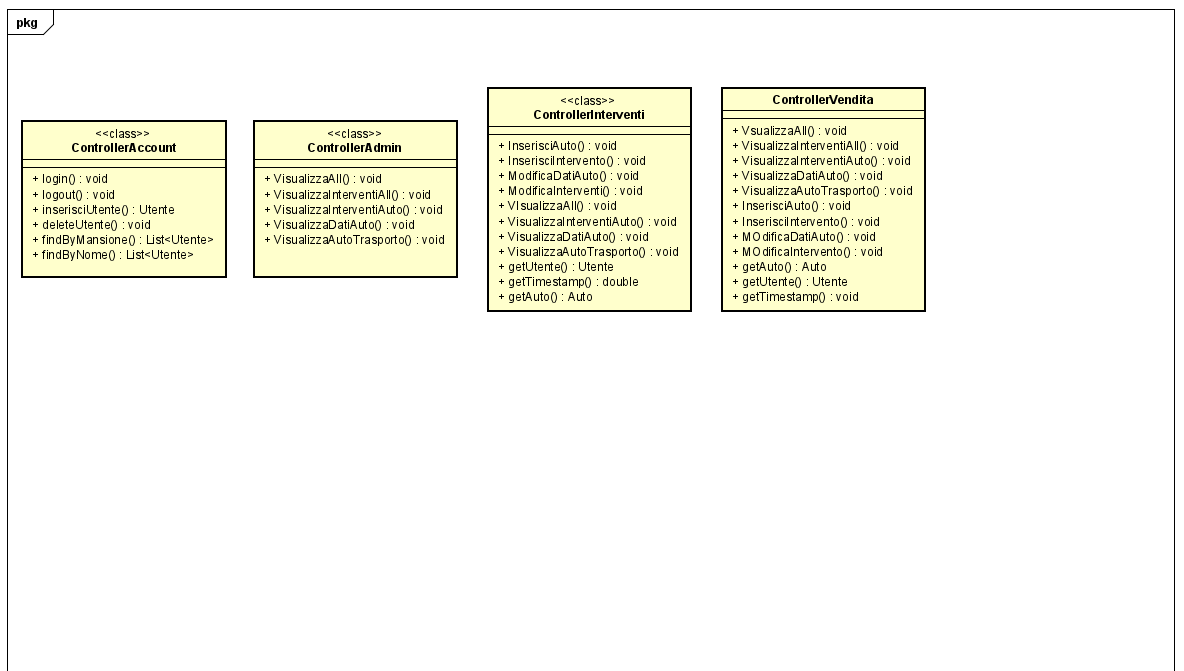


Figura : Iterazione 1 - Diagramma delle classi (interfacce)

### Pseudocodice e analisi complessità tempo

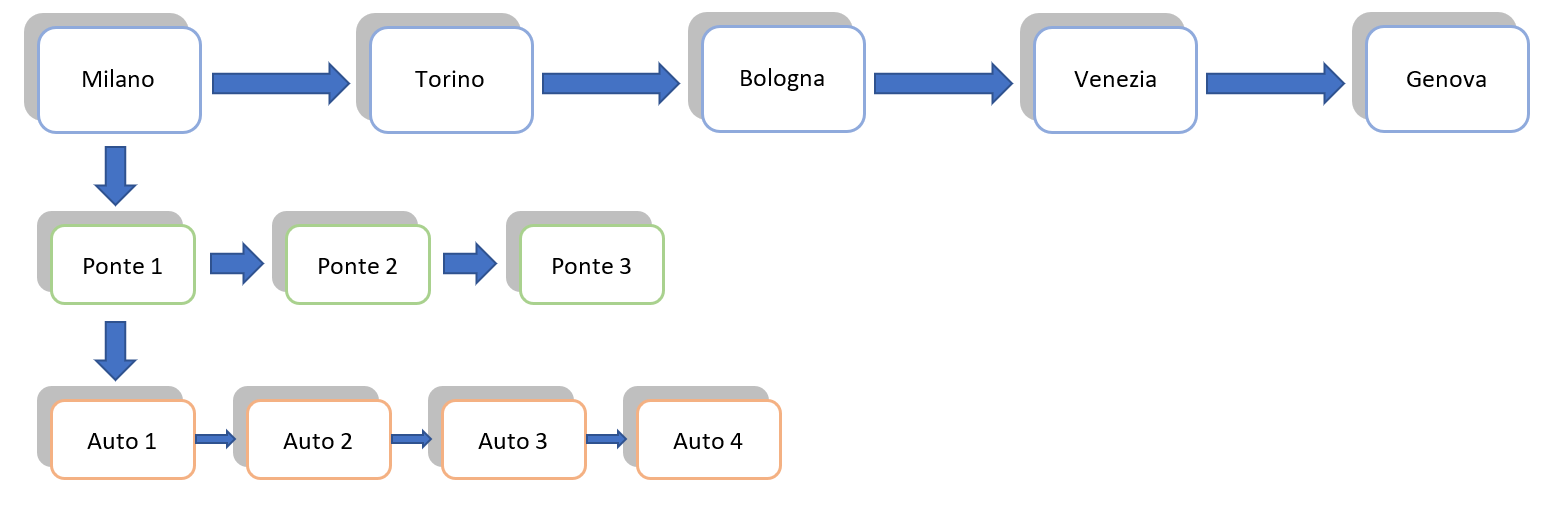
La struttura su cui si basa lo pseudocodice è descritta dalla seguente figura:

Figura : Iterazione 1 - struttura per determinare l'algoritmo

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:

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:

### Analisi statica

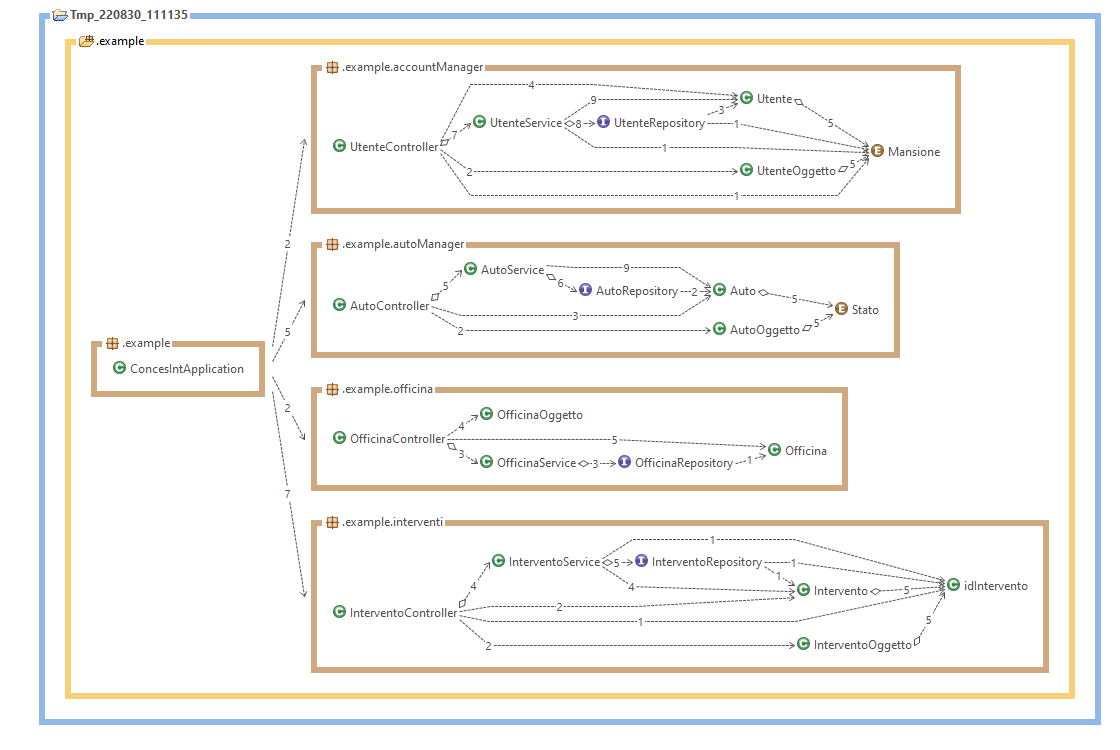
Attraverso l’analisi statica si è voluto verificare che le varie componenti nella prima fase di sviluppo fossero totalmente indipendenti e raccogliessero tutte le classi necessarie alla struttura a microservizi.

Figura : Iterazione 1 - analisi statica

## Iterazione intermedia

In questa iterazione è stato aggiunto l’algoritmo per schedulare le macchine e l’implementazione delle componenti è stata estesa, in modo che nelle ultime iterazioni si dovranno aggiungere 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

Figura : 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 loro dati per poter determinare la schedulazione delle macchine.

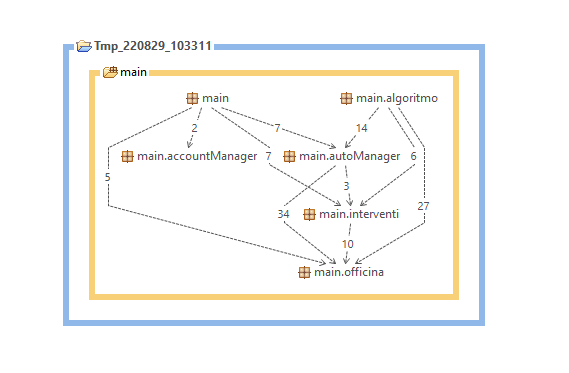


Figura : Iterazione intermedia - analisi statica

### 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.

Immagine che contiene tavolo

Descrizione generata automaticamente

Figura : Iterazione intermedia - analisi dinamica

## Iterazione finale

In questa iterazione sono state aggiunte le interfacce con cui gli utenti utilizzano il programma.

### Analisi requisiti e casi d’uso

I requisiti e i casi d’uso sono rimasti gli stessi delle iterazioni precedenti, quindi sono omessi.

### Design dell’architettura software

Il design dell’architettura software è rimasto il medesimo.

### Analisi statica

### Analisi dinamica

# Guida all’installazione dell’applicazione

# Uso dell’applicazione