

Università degli Studi di Camerino



Relazione Progetto di Ingegneria del Software



SmartBUS

“Tutto passa, tranne l'autobus che stiamo aspettando.”
(Beppe Grillo)



Esaminandi:

*Simone Passaretti
Matteo Cappella
Gianfranco Tribuiani
Giorgio Leonarduzzi*

Sommario

Introduzione	1
Glossario	1
Attori	2
Requisiti	2
Acquisto Biglietti/Abbonamenti	2
Biglietto/Abbonamento Virtuale	2
Stato Della Corsa	2
Calcolo Ritardo	2
Tratte e Orari	3
Confort	3
Servizio Notifiche	3
Casi d'uso	4
Use Case Diagram	4
Registrazione	5
Login	5
VisualizzaOrariTratte	6
AcquistaBigliettiAbbonamenti	6
ControllaConfort	7
VisualizzaNotifiche	7
ControlloValiditàBiglietti	8
Registrazione controllore	8

InvioNotifiche	9	
InvioDati	9	II
Pagamento	10	
Pagamento – sequenza alternativa	10	
 Modelli di analisi	 11	
Activity Diagram	11	
Class Diagram	13	
Sequence Diagram	16	
SeqAcquistaBA	13	
SeqPagamento	19	
SeqInvioDati	20	
SeqLogin	20	
SeqRegistrazione	21	
SeqVisualizzaNotifiche	21	
SeqInvioNotifiche	22	
SeqVisualizzaTratteOrariConfort	23	
 Modelli di progettazione	 24	
Class Diagram	24	
Sequence Diagram	26	
SeqPagamento	26	
SeqInvioDati	26	
SeqAcquistaBA	27	
SeqInvioNotifiche	30	
SeqVisualizzaNotifiche	31	
SeqVisualizzaTratteOrariConfort	32	
SeqRegistrazione	33	

SeqLogin	34	
Package Diagram	35	III
Implementazione del prototipo	37	
Visualizza Tratte e Orari	38	
Visualizza Notizie	38	
Acquisto Biglietti e Abbonamenti	38	
Pagina Amministratore	39	
Tecnologie utilizzate	39	
Struttura del percorso dei file	40	
Classi Java	41	
Javascript	42	
Struttura del database	42	
Tabelle	42	
Utente	42	
Tratta	43	
Corsa	43	
Orario	43	
Fermata	44	
Compagnia	44	
Città	44	
Notizia	45	
Biglietti	45	
<i>Extraurbano</i>	45	
<i>Urbano</i>	45	
Abbonamenti	46	

<i>Urbano</i>	46	
<i>Extraurbano</i>	46	IV
Query	46	

1. Introduzione

Si vuole realizzare un'applicazione chiamata SmartBus che consente a qualsiasi utente di acquistare biglietti online per il trasporto pubblico, utilizzando il proprio smartphone o comodamente da casa, via web attraverso il proprio terminale. Con questo strumento, qualsiasi utente registrato può acquistare comodamente il biglietto per la tratta scelta oppure può optare per l'acquisto di un intero abbonamento per la tratta desiderata. Oltre alla scelta di acquisto, il cliente ha anche a disposizione servizi ed informazioni su tutto ciò che riguarda il viaggio. Essi riguardano la possibilità di visualizzare sul proprio cellulare o pc lo stato della corsa a cui si è interessati; ad esempio il cliente può verificare se la corsa ha subito eventuali ritardi alle fermate ed inoltre riceve anche la quantità di ritardo accumulato in termini di tempo. Un altro servizio molto utile consiste nel poter visualizzare in tempo reale sia orari sia le tratte desiderati. Una caratteristica novità dell'applicazione, è che quest'ultima mette a disposizione al cliente anche un livello di comfort corrispondente ad ogni specifico mezzo di trasporto, in tempo reale grazie a dei sensori inseriti sui mezzi stessi. L'utente, infine, viene avvisato, attraverso l'invio di apposite notifiche, del verificarsi di situazioni che potrebbero compromettere il viaggio da intraprendere come ad esempio incidenti o scioperi.

2. Glossario

TERMINE	DEFINIZIONE	SINONIMI
ABBONAMENTO (Cartaceo)	Contratto per cui, pagando una determinata somma, si può, per un tempo determinato, usufruire del servizio di trasporto pubblico	TITOLO DI VIAGGIO
ABBONAMENTO ELETTRONICO	Versione elettronica dell'abbonamento cartaceo acquistabile online	ABBONAMENTO VIRTUALE
BIGLIETTO (Cartaceo)	Piccolo stampato rilasciato come contrassegno del diritto di usufruire di un pubblico servizio. (Ferroviario, tranviario)	TICKET, TITOLO DI VIAGGIO
BIGLIETTO ELETTRONICO	Versione elettronica del biglietto cartaceo acquistabile online.	TICKET ELETTRONICO, TITOLO DI VIAGGIO ELETTRONICO, BIGLIETTO VIRTUALE
CORSA	Viaggio effettuato lungo una tratta, a una determinata ora da un autobus	"NESSUNO"
TRATTA	Tragitto, identificato da un numero univoco, effettuato più volte in uno stesso giorno ad orari differenti da un autobus	PERCORSO, ITINERARIO, GIRO

3. Attori

1. Utente: l'utente è colui che usa i servizi offerti dall'applicazione SmartBus.
2. Controllore: il controllore ha il compito di verificare la validità dei biglietti o abbonamenti attraverso il riconoscimento del codice qr code.
3. Amministratore: gestore dell'applicazione, ha il compito di inserire le Notifiche di avviso nella bacheca dell'applicazione su cui l'utente può visualizzarle.

4. Requisiti

ID: R_01

Nome: Acquisto Biglietti/Abbonamenti

Descrizione: Il Sistema deve garantire all'utente finale, tramite un'apposita interfaccia multipiattaforma, la possibilità di acquistare un biglietto per il pullman.

Motivazione: Agevolare e Migliorare gli spostamenti che le persone devono compiere per adempiere agli impegni quotidiani.

Criterio di Valutazione: Sistema stabile e sicuro, reperibile 24/7.

Priorità: Alta

Sorgente: il Team di Sviluppo

ID: R_02

Nome: Biglietto/Abbonamento Virtuale

Descrizione: Il sistema deve produrre un documento digitale legalmente riconosciuto come Titolo di Viaggio, comprensivo di codice Qr 29x29 approvato dallo standard ISO/IEC 18004:2006.

Motivazione: Pratico per il cliente grazie alla possibile visualizzazione digitale, per il controllore grazie alla rapidità con cui verificare idoneità genuinità, ecologicamente sostenibile.

Criterio di Valutazione: Corretta generazione del biglietto digitale da parte del software adibito.

Priorità: Alta

Sorgente: il Team di Sviluppo

ID: R_03

Nome: Stato Della Corsa

Descrizione: Il sistema può permettere all'utente – salvo disguidi – di visualizzare a video tramite scritta l'esatta via in cui il mezzo si trova.

Motivazione: Aiutare il Cliente a capire se dovrà attendere la corsa successiva o attendere il giusto quantitativo di tempo per l'arrivo del mezzo.

Criterio di Valutazione: Il sistema di posizionamento globale (GPS) è attivo e trasmette in modo corretto i dati al sistema, il sistema SmartBus è pienamente operativo.

Priorità: Media

Sorgente: il Team di Sviluppo

ID: R_04

Nome: Calcolo Ritardo

Requisito: Il sistema potrebbe, in caso di necessità e corretto funzionamento dell'apparato GPS, calcolare e visualizzare sull'Interfaccia utente il possibile ritardo accumulato dal mezzo.

Motivazione: Aiutare il Cliente a capire se dovrà attendere la corsa successiva o attendere il giusto quantitativo di tempo per l'arrivo del mezzo.

Criterio di Valutazione: Il sistema di posizionamento globale (GPS) è attivo e trasmette in modo corretto i dati al sistema. Il sistema SmartBus è pienamente operativo.

Priorità: Bassa

Sorgente: Il Team di Sviluppo

ID: R_05

Nome: Tratte e Orari

Descrizione: Il sistema deve provvedere alla visualizzazione, mediante apposita interfaccia, della lista delle tratte e delle corse salvate nel database del sistema, fornite dalla Compagnia di Trasporti

Motivazione: Agevolare e Migliorare l'organizzazione della propria giornata.

Criterio di Valutazione: Sistema stabile e sicuro, reperibile 24/7.

Priorità: Alta

Sorgente: il Team di Sviluppo

ID: R_06

Nome: Comfort

Descrizione: Il sistema può – salvo disfunzioni- visualizzare il livello di comfort calcolato da valori percentuali forniti dai sensori montati on board (umidità, temperatura, rumorosità, disponibilità posti a sedere, qualità aria...).

Motivazione: Garantire un servizio all'altezza a persone bisognose o particolarmente esigenti.

Criterio di Valutazione: Il sistema sensoristico on board è attivo e trasmette in modo corretto i dati al sistema, il sistema SmartBus è pienamente operativo.

Priorità: Media

Sorgente: il Team di Sviluppo

ID: R_07

Nome: Servizio Notifiche

Descrizione: Il sistema deve tenere gli utenti sempre aggiornati su scioperi, disservizi o altre comunicazioni di servizio.

Motivazione: Agevolare e Migliorare l'organizzazione della propria giornata.

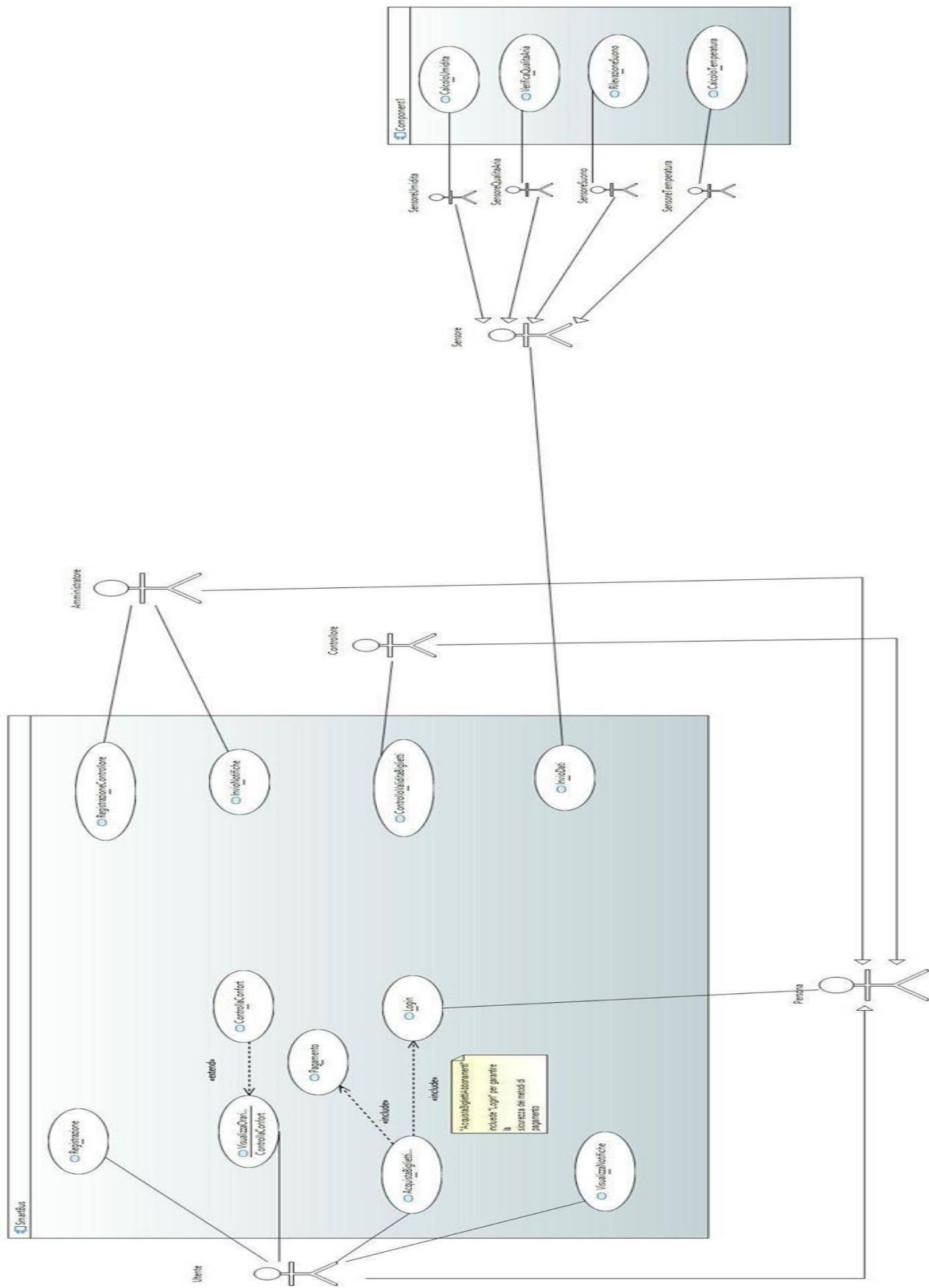
Criterio di Valutazione: Sistema stabile e sicuro, reperibile 24/7 e notifiche correttamente inviate dall'amministratore.

Priorità: Alta

Sorgente: il Team di Sviluppo

4.1 Casi d'Uso

4



Caso d'uso: Registrazione
ID:1
Breve descrizione: Il sistema crea un nuovo account per l'Utente
Attori primari: Utente
Attori secondari: Nessuno.
Precondizioni: Nessuna.
Sequenza degli eventi principale: 1. Il caso d'uso inizia quando l'Utente seleziona "Crea nuovo account Utente" 2. Fintantochè le informazioni dell'Utente non sono valide 2.1. Il sistema chiede all'Utente di inserire le sue informazioni tra cui nome, cognome, indirizzo email, password e password di conferma 2.2. Il sistema valida le informazioni dell'Utente 3. Il sistema crea un nuovo account per l'Utente
Postcondizioni: 1. Un nuovo account per l'Utente è stato creato
Sequenze degli eventi alternative: Nessuna.

Caso d'uso: Login
ID:2
Breve descrizione: Il sistema autentica l'Utente, il Controllore o l'Amministratore
Attori primari: Utente, Controllore, Amministratore
Attori secondari: Nessuno.
Precondizioni: 1. L'Utente, il Controllore e l'Amministratore devono essere registrati.
Sequenza degli eventi principale: 1. Il caso d'uso inizia quando l'Utente, il Controllore o l'Amministratore seleziona "Login" 2. Fintantochè nome e password non sono valide 2.1 Il sistema chiede all'Utente, al Controllore o all'Amministratore di inserire il nome dell'Utente e la password 2.2 Il sistema valida il nome dell'Utente e la password 3. Il sistema autentica l'Utente, il Controllore o l'Amministratore
Postcondizioni: 1. L'Utente, il Controllore o l'Amministratore devono essere autenticati
Sequenze degli eventi alternative: Nessuna.

Caso d'uso: VisualizzaOrariTratte
ID:3
Breve descrizione: Il sistema mostra orari e tratte dei bus.
Attori primari: Utente
Attori secondari: Nessuno.
Precondizioni: Nessuna.
Sequenza degli eventi principale: 1. Il caso d'uso inizia quando l'Utente seleziona "Visualizza orari e tratte" 2. L'Utente seleziona la tratta 3. Il sistema mostra gli orari relativi alla tratta
Postcondizioni: Nessuna.
Sequenze degli eventi alternative: Nessuna.

Caso d'uso: AcquistaBigliettiAbbonamenti
ID:4
Breve descrizione: L'Utente acquista un biglietto o un abbonamento
Attori primari: Utente
Attori secondari: Nessuno.
Precondizioni: 1. L'Utente deve essere autenticato.
Sequenza degli eventi principale: 1. Il caso d'uso inizia quando l'Utente seleziona "Acquista biglietto o abbonamento" 2. il sistema geolocalizza l'Utente, o l'Utente stesso sceglie manualmente la città 3. L'Utente seleziona la compagnia tra quelle mostrate dal sistema per la <u>città</u> selezionata 4. Se l'Utente seleziona "Acquista biglietto" 4.1. L'Utente seleziona la tratta tra quelle disponibili 5. Se l'Utente seleziona "Acquista abbonamento" 5.1. L'Utente seleziona il tipo di abbonamento tra quelli che la compagnia mette a disposizione 6. Include (Pagamento)
Postcondizioni: Il biglietto o l'abbonamento è stato acquistati
Sequenze degli eventi alternative: Nessuna

Caso d'uso: ControllaConfort
ID:5
Breve descrizione: L'Utente visualizza il grado di confort di uno specifica corsa
Attori primari: Utente
Attori secondari: Nessuno.
Precondizioni: Nessuna.
Sequenza degli eventi principale: 1. Il caso d'uso inizia quando l'Utente seleziona "Controlla confort" 2. L'Utente seleziona la tratta 3. L'Utente seleziona la corsa 4. Il sistema mostra il grado di confort
Postcondizioni: Nessuna.
Sequenze degli eventi alternative: Nessuna.

Caso d'uso: VisualizzaNotifiche
ID:6
Breve descrizione: L'Utente visualizza le notifiche
Attori primari: Utente
Attori secondari: Nessuno.
Precondizioni: 1.L'Utente deve essere autenticato
Sequenza degli eventi principale: 1. Il caso d'uso inizia quando l'Utente seleziona "VisualizzaNotifiche" 2. Il sistema geolocalizza l'Utente o l'Utente seleziona la città 3. Il sistema mostra le notifiche per la città selezionata
Postcondizioni: Nessuna.
Sequenze degli eventi alternative: Nessuna.

8

Caso d'uso: ControlloValiditàBiglietti	
ID:7	
Breve descrizione:	
Il Controllore controlla la validita' dei biglietti o degli abbonamenti	
Attori primari:	
Controllore	
Attori secondari:	
Nessuna	
Precondizioni:	
1.Il Controllore deve essere autenticato	
Sequenza degli eventi principale:	
1. Il caso d'uso inizia quando il Controllore seleziona "Controlla validita' biglietti"	
2. Il Controllore inserisce il codice univoco o scannerizza il qr code che identifica il biglietto o l'abbonamento in maniera univoca	
3. Se il biglietto o abbonamento è valido	
3.1 Il sistema mostra "Valido"	
4. Se il biglietto o abbonamento non è valido	
4.1 Il sistema mostra "Non valido"	
Postcondizioni:	
Nessuna.	
Sequenze degli eventi alternative:	
Nessuna.	

Caso d'uso: RegistrazioneControllore	
ID:8	
Breve descrizione:	
Il sistema crea un nuovo account per il Controllore	
Attori primari:	
Amministratore	
Attori secondari:	
Controllore	
Precondizioni:	
1.L'Amministratore deve essere autenticato	
Sequenza degli eventi principale:	
1. Il caso d'uso inizia quando l'Amministratore seleziona "Crea nuovo account Controllore"	
2. Fintantochè le informazioni del Controllore non sono valide	
2.1. Il sistema chiede all'Amministratore di inserire le informazioni del Controllore tra cui nome, cognome, indirizzo email, password e password di conferma	
2.2. Il sistema valida le informazioni del Controllore	
3. Il sistema crea un nuovo account per il Controllore	
Postcondizioni:	
1. Un nuovo account per il Controllore è stato creato	
Sequenze degli eventi alternative:	
Nessuna.	

9

Caso d'uso: InvioNotifiche	
ID:9	
Breve descrizione:	L'Amministratore scrive e invia le notifiche
Attori primari:	Amministratore
Attori secondari:	Nessuno.
Precondizioni:	1.L'Amministratore deve essere autenticato 2.L'Amministratore scrive la notifica 3.Il sistema pubblica la notifica
Sequenza degli eventi principale:	1. Il caso d'uso inizia quando l'Amministratore seleziona "Nuova notifica" 2. L'Amministratore scrive la notifica 3. Il sistema pubblica la notifica
Postcondizioni:	1. La notifica è stata pubblicata
Sequenze degli eventi alternative:	Nessuna.

Caso d'uso: InvioDati	
ID:10	
Breve descrizione:	Il sistema riceve dati dai sensori
Attori primari:	Sensore
Attori secondari:	Nessuno.
Precondizioni:	Nessuna.
Sequenza degli eventi principale:	1. Il caso d'uso inizia in qualsiasi momento 2. Il sistema riceve i dati dai sensori 3. Il sistema immagazzina i dati
Postcondizioni:	Nessuna.
Sequenze degli eventi alternative:	Nessuna.

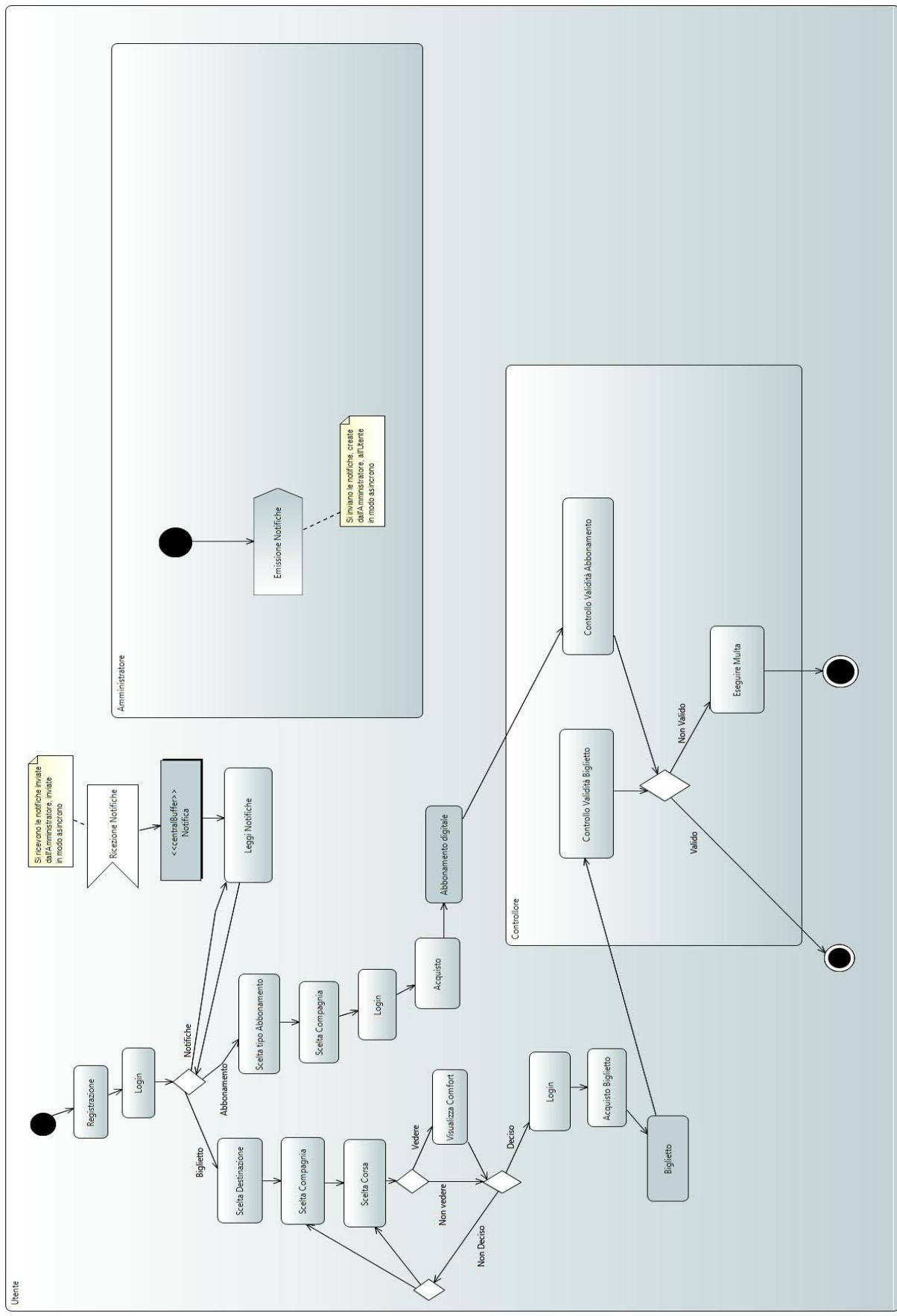
Caso d'uso: Pagamento
ID:11
Breve descrizione:
L'utente effettua il pagamento dell'acquisto di un biglietto o di un abbonamento
Attori primari:
Utente
Attori secondari:
Nessuno
Precondizioni:
L'utente ha acquistato un biglietto o un abbonamento
Sequenza degli eventi principale:
<ol style="list-style-type: none"> 1. Il caso d'uso inizia quando l'Utente seleziona "Pagamento" 2. Include (Login) 3. L'Utente sceglie il metodo di pagamento tra quelli disponibili 4. Il sistema controlla che il pagamento sia andato a buon fine
Postcondizioni:
Il pagamento è andato a buon fine.
Sequenza degli eventi alternative:
PagamentoNonRiuscito

Sequenza degli eventi alternativa: Pagamento: PagamentoNonRiuscito
ID:11.1
Breve descrizione:
Il sistema informa che l'operazione di pagamento non è andata a buon fine
Attori primari:
Utente
Attori secondari:
Nessuno.
Precondizioni:
1. Il pagamento non è andato a buon fine
Sequenza degli eventi alternativa:
<ol style="list-style-type: none"> 1. La sequenza degli eventi alternativa inizia dopo il passo 4 della sequenza degli eventi principali 2. Il sistema informa l'Utente che l'operazione di pagamento non è andata a buon fine
Postcondizioni:
Nessuna.

5. Modelli di Analisi

5.1 Activity Diagram

11



Questo particolare tipo di diagramma ci ha consentito, in fase di analisi, di modellare il funzionamenti dell'intero progetto secondo un flusso di azioni ordinato e sequenziale.

12

Partendo dal contesto “Utente” descrive che l’utilizzatore dovrà registrarsi ed effettuare l’autenticazione prima di poter scegliere di compiere una qualsiasi azione tra:

- acquisto biglietto
- acquisto abbonamento
- lettura notifiche

Nel caso scelga di acquistare un biglietto il sistema gli proporrà di scegliere una destinazione, una compagnia di trasporti e la corsa specifica. Gli verrà poi proposto di visualizzare il grado del confort calcolato all’interno della corsa scelta e di confermare l’acquisto. In caso di mancata conferma potrà scegliere di cambiare compagnia e/o corsa. In caso di conferma dovrà rieffettuare l’autenticazione e poi seguire la procedura di acquisto per ricevere l’oggetto “Biglietto”.

Nel caso scelga di acquistare un abbonamento dovrà scegliere il tipo di abbonamento desiderato (urbano/extraurbano) e la compagnia a cui appoggiarsi. Se conferma l’acquisto dovrà rieffettuare il login e seguire la procedura di acquisto per ricevere l’oggetto “Abbonamento”.

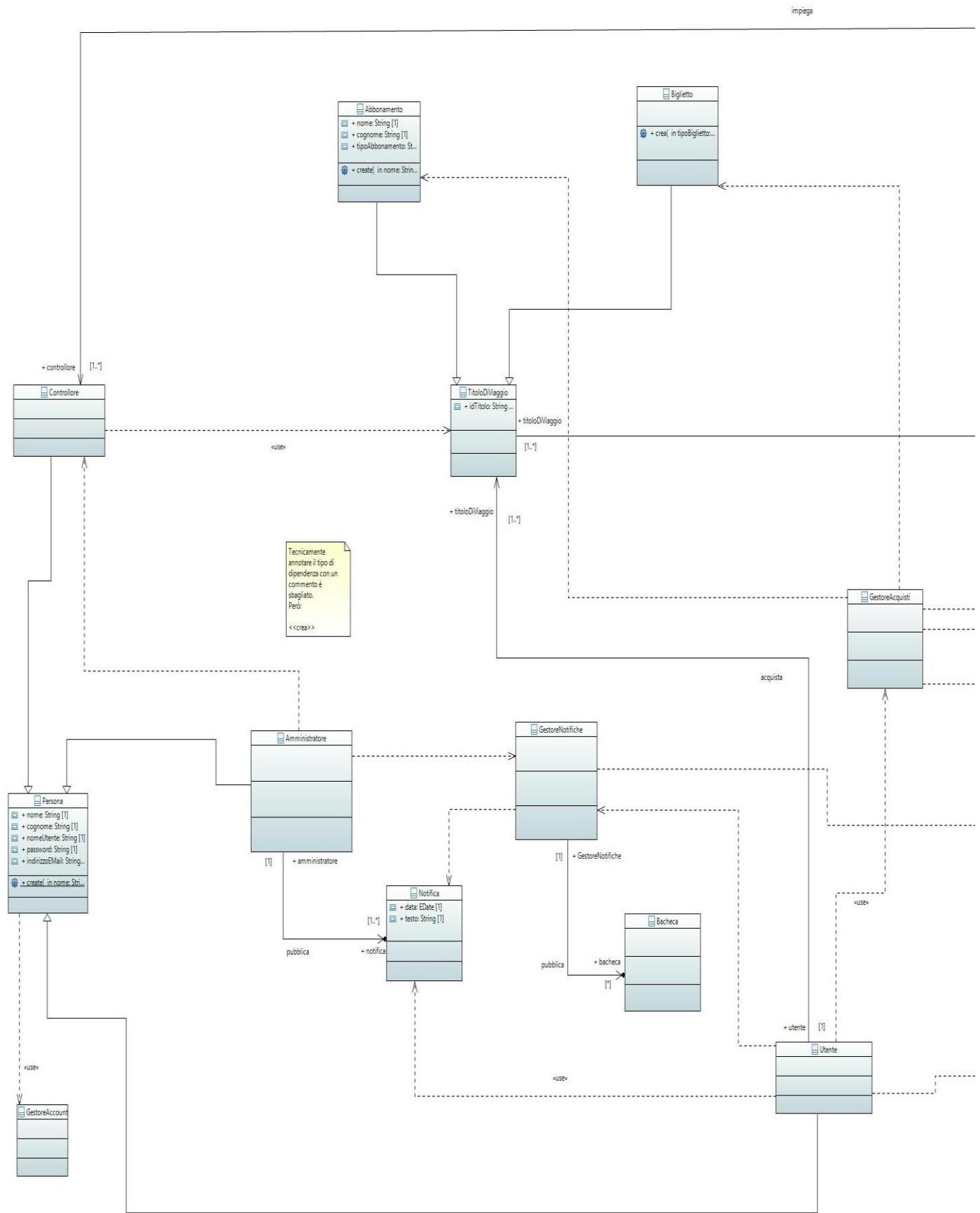
Nel caso scelga di leggere le notifiche dovrà accedere alla bacheca in cui l’amministratore pubblica le varie comunicazioni.

Il contesto “Amministratore” descrive come l’amministratore del sistema emetta le notifiche all’interno della bacheca sopra indicata.

Il contesto “Controllore” descrive il comportamento che il controllore deve effettuare partendo dalla verifica di validità dei titoli di viaggio e arrivando a scegliere se effettuare o meno una multa.

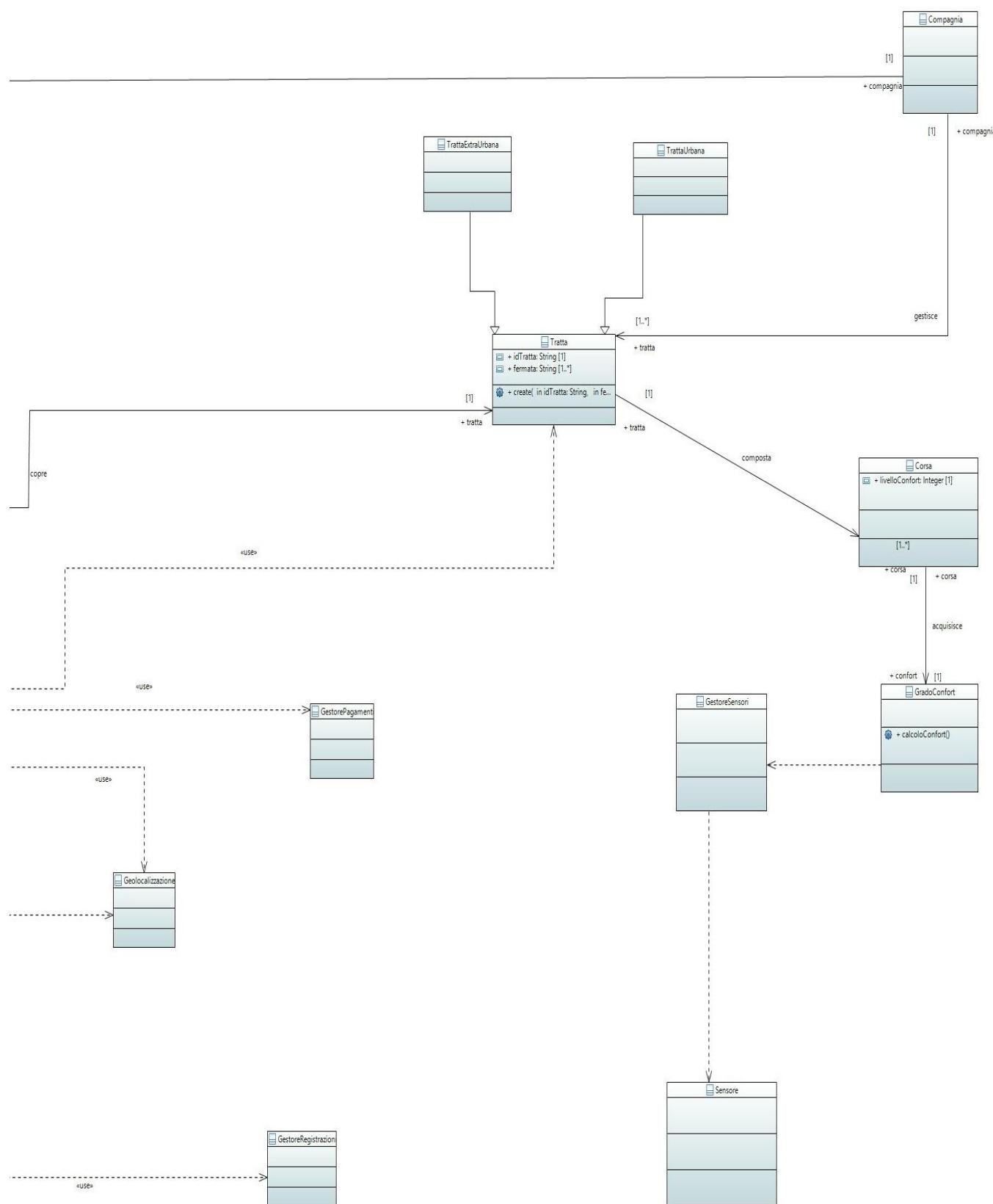
5.2 Class Diagram – 1/2

13



5.2 Class Diagram - 2/2

14



Questo diagramma è stato utilizzato per descrivere, grazie alle sue proprietà di sintesi e schematizzazione, l'intero dominio del nostro progetto, visualizzando per esteso tutti i possibili concetti correlati (sottoforma di classi) e tracciando tra di essi collegamenti semantici.

Iniziando la lettura di tale schema dalla sua sinistra osserviamo la classe "Persona". Essa viene generalizzata in tre distinte categorie semantiche del concetto di "persona": Utente, Amministratore e Controllore. È collegata mediante lo stereotipo di dipendenza "«usa»" alla classe "GestoreAccount" - quest'ultima classe è l'idealizzazione concettuale di un meccanismo di autenticazione che avrà il sistema.

La generalizzazione "Amministratore" è collegata mediante dipendenza "«crea»" alla classe "Controllore", da un'associazione "pubblica" alla classe "Notifica" e mediante la dipendenza "«usa»" alla classe "GestoreNotifiche": tale classe è l'idealizzazione concettuale di un meccanismo di gestione software dell'emissione delle notifiche che l'amministratore potrà pubblicare.

La generalizzazione "Utente" ha dipendenze di tipo "«usa»" con le classi: "Notifica", "GestoreRegistrazioni", "GestoreNotifica", "GestoreAcquisti"; Tali classi "gestore" sono idealizzazioni concettuali di meccanismi rispettivamente a: la registrazione dei dati dell'utente utilizzatore, la gestione suddetta delle notifiche e alla sequenza di azioni procedurali, legali ed economiche legate all'acquisto di un titolo di viaggio. Possiede inoltre un'associazione "acquista" con la classe "TitoloDiViaggio".

La generalizzazione "Controllore" ha una dipendenza "«usa»" con la classe "TitoloDiViaggio".

La classe "GestoreNotifiche" ha una dipendenza "«usa»" con la classe "Geolocalizzazione" e "Notifica" ed un'associazione con la classe "Bacheca" – la quale non ha alcun altro collegamento.

La classe "Compagnia" è la classe che idealizza non la compagnia fornitrice del nostro servizio bensì una compagnia di trasporti. È collegata con l'associazione "impiega" alla classe "Controllore" e con l'associazione "gestisce" alla classe "Tratta".

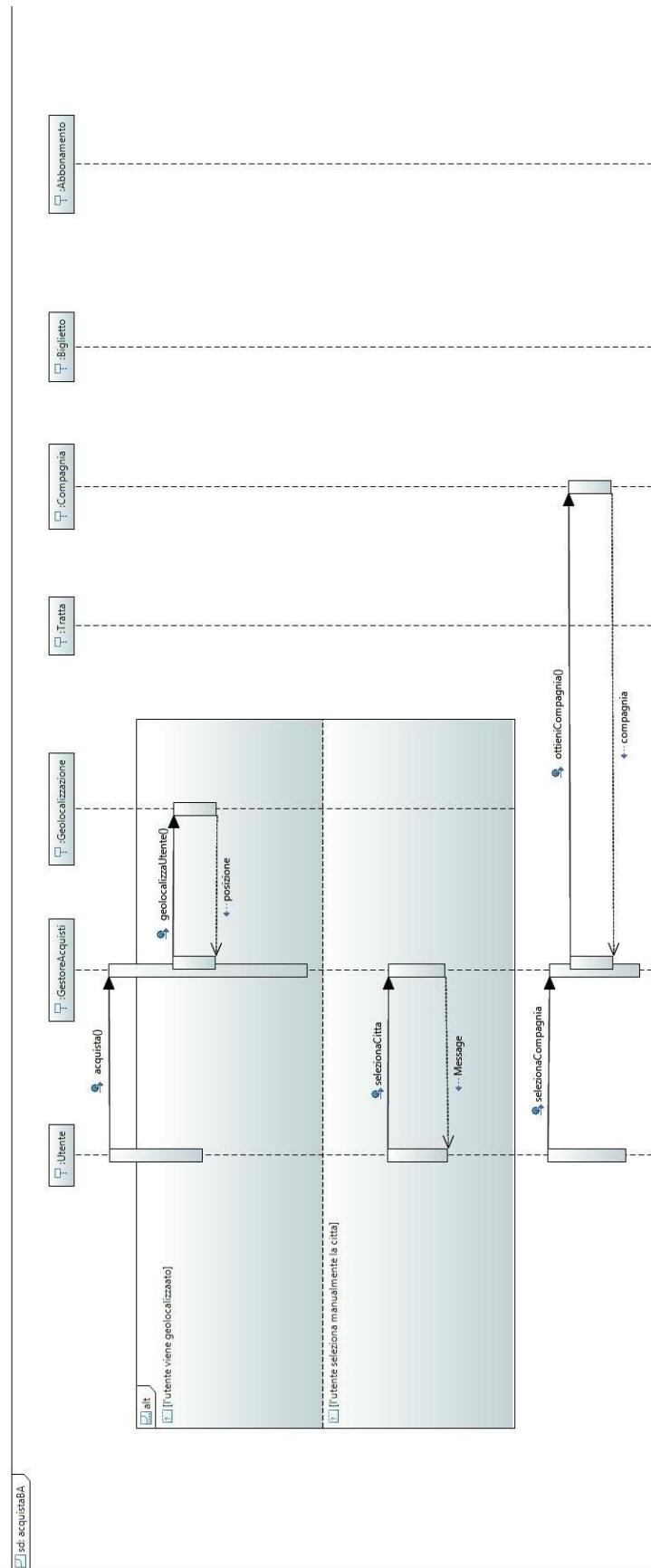
La classe "Tratta" generalizza i concetti di "tratta urbana" e "tratta extraurbana" ed è collegata con un'associazione "composta" alla classe "Corsa" e ad un'associazione "copre" alla classe "TitoloDiViaggio": quest'ultima viene generalizzata da "Abbonamento" e "Biglietto".

La classe "GestoreAcquisti" ha dipendenze di tipo "«usa»" con: "Abbonamento", "Biglietto", "Tratta", "Geolocalizzazione" e "GestorePagamenti". Queste ultime due non hanno altri collegamenti se non quelli citati in precedenza.

La classe "Corsa" presenta una sola associazione "acquisisce" con la classe "GradoConfort", la quale ha una dipendenza con "GestoreSensori" e a cascata con la classe "Sensore". Queste ultime quattro classi non hanno altri collegamenti se non i suddetti.

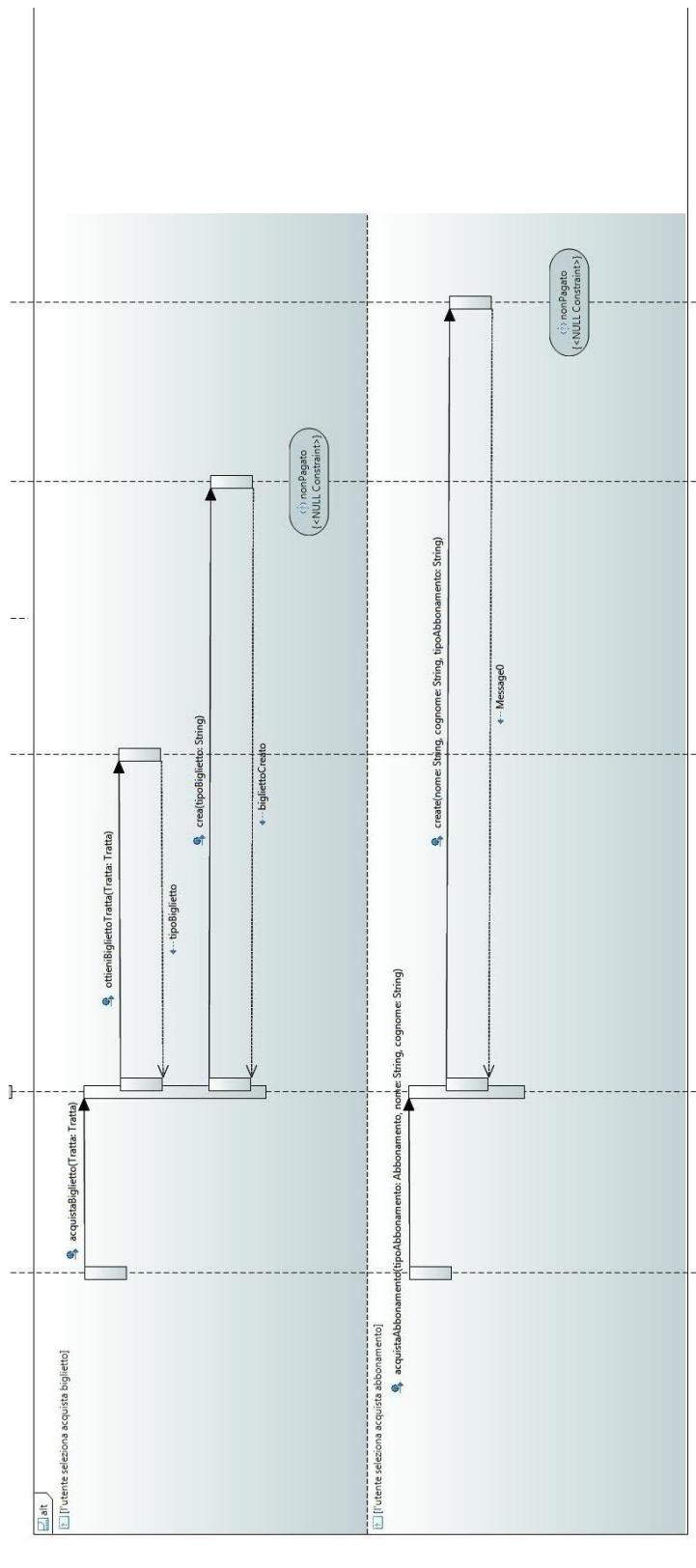
5.3.1 Sequence Diagram – SeqAcquistaBA – 1/3

16



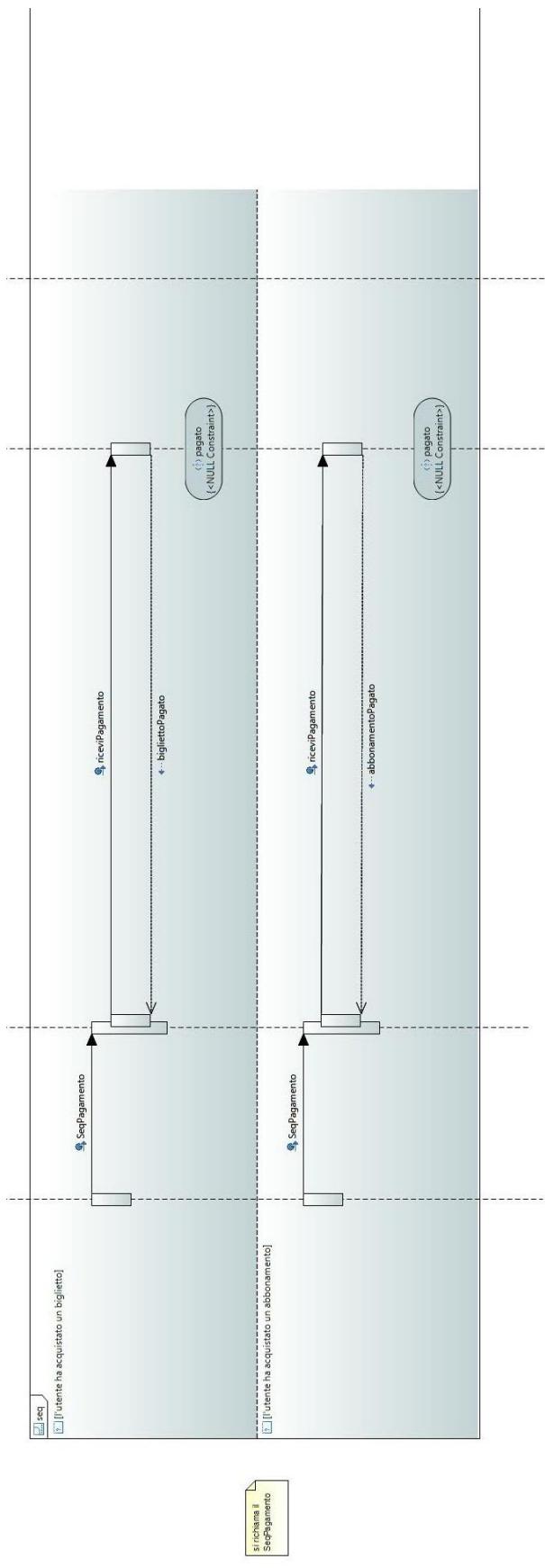
5.3.1 Sequence Diagram – SeqAcquistaBA – 2/3

17



5.3.1 Sequence Diagram – SeqAcquistaBA – 3/3

18



Questo tipo di diagramma consente di visualizzare in modo chiaro, conciso e temporizzato le iterazioni specifiche tra ogni parte (classe) del sistema coinvolta in una determinata azione. Le varie parti coinvolte hanno una linea verticale tratteggiata che segue un'ordinata immaginaria segnante l'avanzare del tempo (“linea di vita”) e dei piccoli rettangoli annessi a tale tratteggio che segnano la durata di una determinata azione.

Nel caso di questo diagramma si va a descrivere la sequenza di azioni che intercorrono nell’acquisto da parte di un utente di un biglietto o di un abbonamento.

La sequenza inizia con l’utente che richiede l’acquisto di un titolo di viaggio al gestore acquisti.

Si accede ora ad un’area condizionale (“frammenti combinati”), denotata dall’operatore “alt”, in cui vengono separati i due casi possibili secondo cui l’utente può scegliere di essere geolocalizzato o scegliere manualmente la città in cui poter poi spendere il titolo di viaggio.

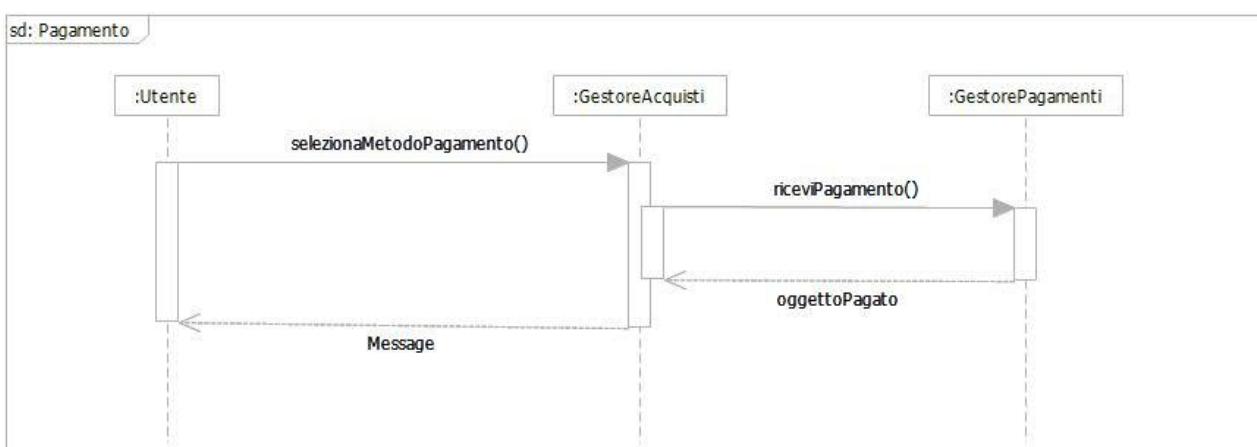
Viene richiesto ora all’utente di selezionare la compagnia i trasporti e il gestore acquisti di conseguenza richiede e riceve la scheda specifica relativa alla compagnia scelta.

Si accede in un’altra area “alt” in cui si esplorano in due casi secondo cui un utente può comperare un biglietto oppure un abbonamento.

Nel caso di un biglietto l’utente selezionerà la tratta desiderata e successivamente invierà al gestore acquisti la richiesta di acquisto. Il gestore tramite un’interrogazione richiederà il tipo (urbano/extraurbano) di biglietto previsto per la tratta scelta. Chiederà poi alla classe “Biglietto” di generare una sua istanza “biglietto”, il cui stato sarà impostato su “nonPagato”. Nel caso di un abbonamento all’utente verrà richiesto di specificare la durata dell’abbonamento (annuale/mensile/settimanale/...), il relativo tipo (urbano/extraurbano) e i suoi dati anagrafici. Il gestore creerà poi una istanza della classe “Abbonamento” avente lo stato anch’esso impostato su “nonPagato”.

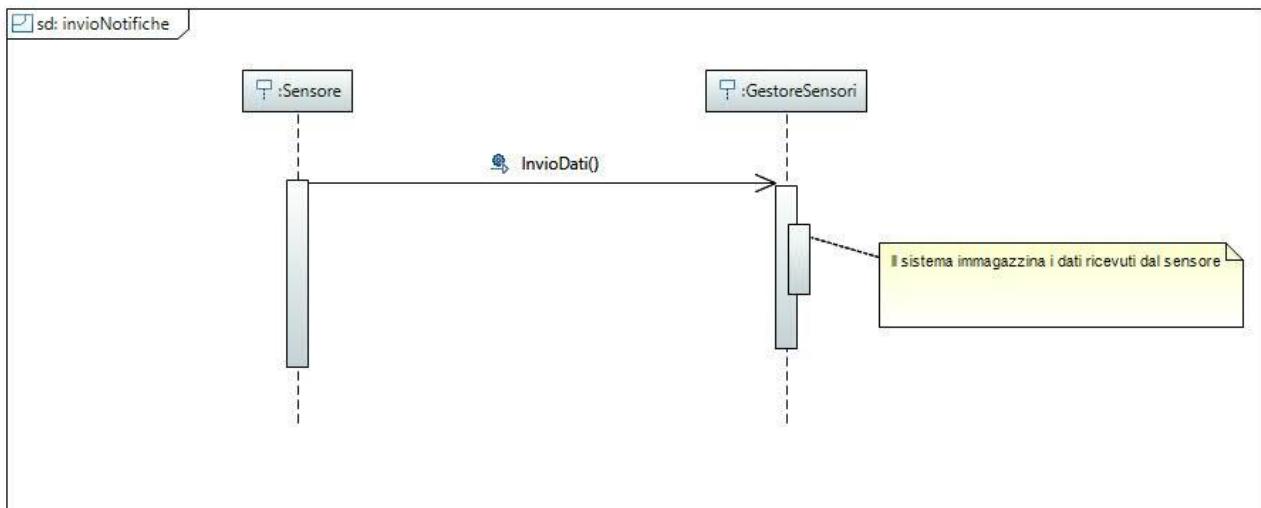
L’ultima parte del diagramma descrive il principio di funzionamento secondo cui si basa la serie di azioni riguardanti il pagamento. Tale insieme di azioni è richiamato dal Sequence Diagram – SeqPagamento, come anche riportato nella nota a margine allegata.

5.3.2 Sequence Diagram – SeqPagamento



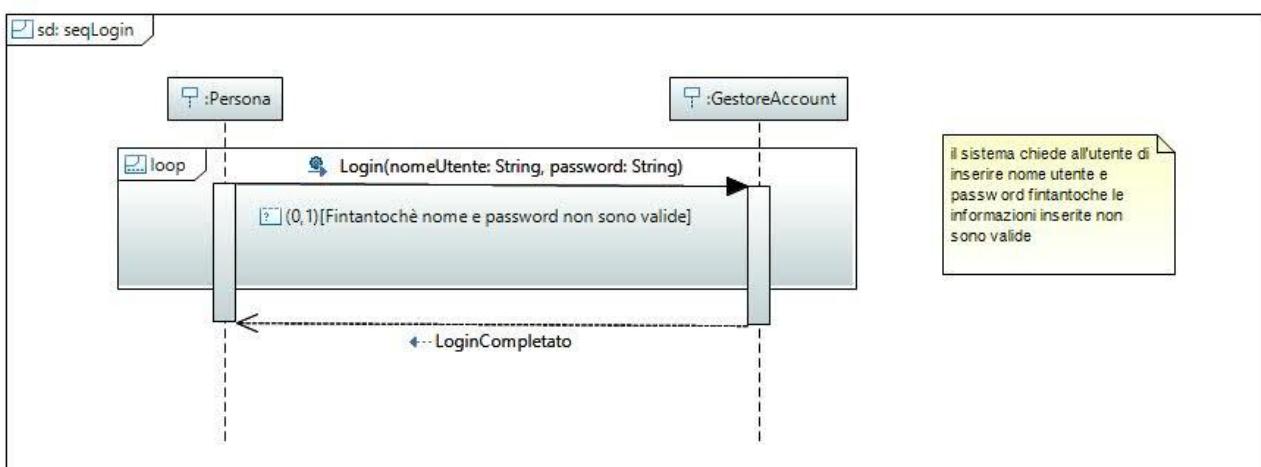
L'utente in questo diagramma avrà a disposizione una serie di metodi di pagamento tra cui sceglierà e invierà la richiesta al gestore correlato. Il gestore invierà un comando con il quale il gestore dei pagamenti potrà portare a termine il pagamento e modificare lo stato dell'oggetto acquistato in "pagato". L'utente riceverà quindi la conferma dell'avvenuto pagamento.

5.3.3 Sequence Diagram – SeqInvioDati



In questo diagramma si descrive come un'entità sensore invii i dati raccolti ad un gestore adibito, il quale li immagazzinerà per un successivo utilizzo.

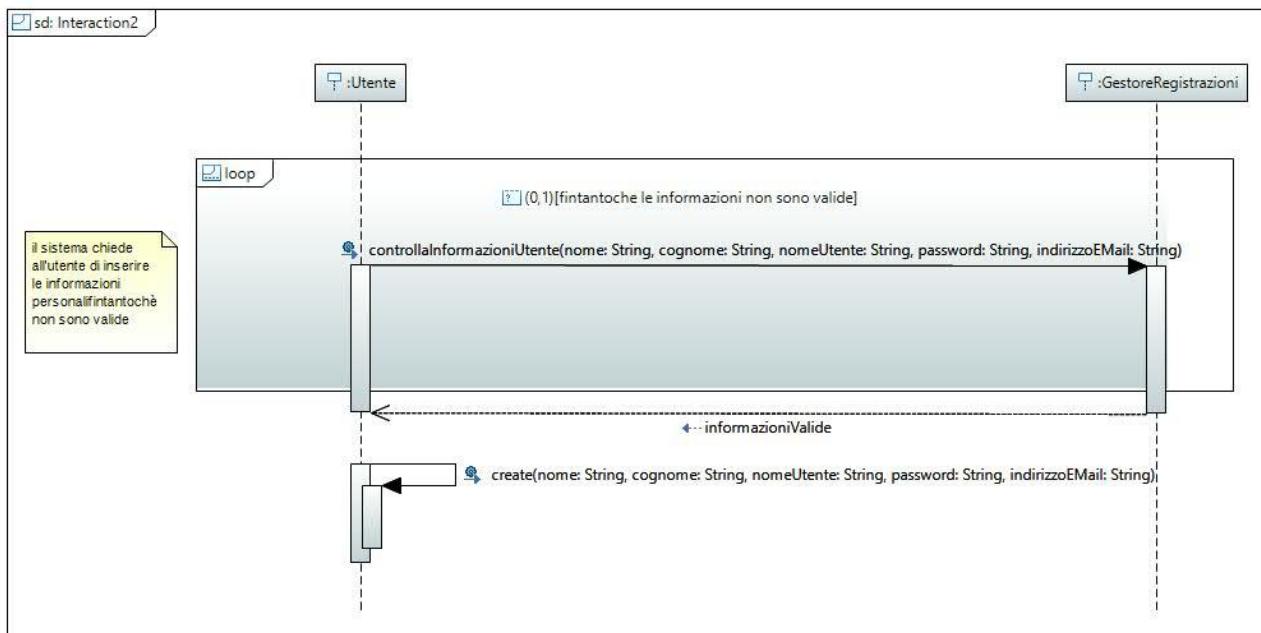
5.3.4 Sequence Diagram – SeqLogin



In questo diagramma la generica persona richiederà al gestore degli account di essere autenticata e ciò avverrà solo se le informazioni inserite saranno valide. Si noti che tale azione avviene in un area denominata "loop" e che indica proprio il requisito sopra citato della correttezza dei dati.

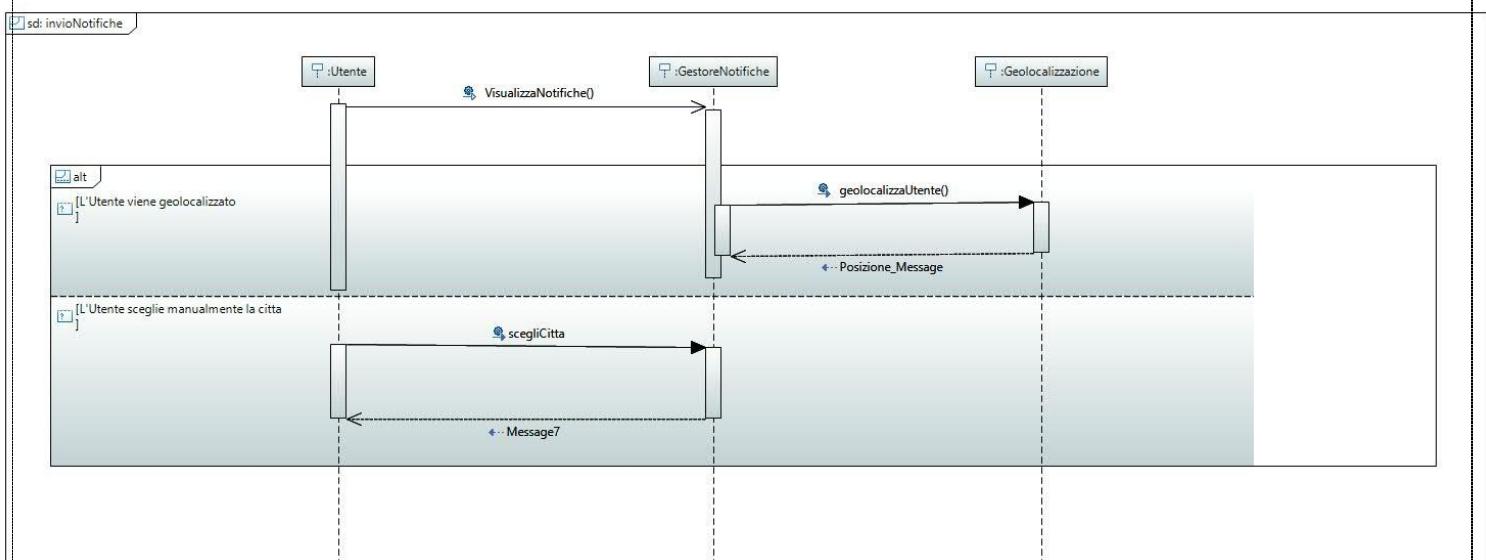
5.3.5 Sequence Diagram – SeqRegistrazione

21



Il diagramma sopra allegato mostra come l'azione di registrazione sia vincolata da un'area di "loop" la cui condizione di uscita impone di inserire dei dati personali validi e corretti. Al seguito di tale immissione il gestore delle registrazioni manderà una nota di feedback. L'utente verrà poi creato.

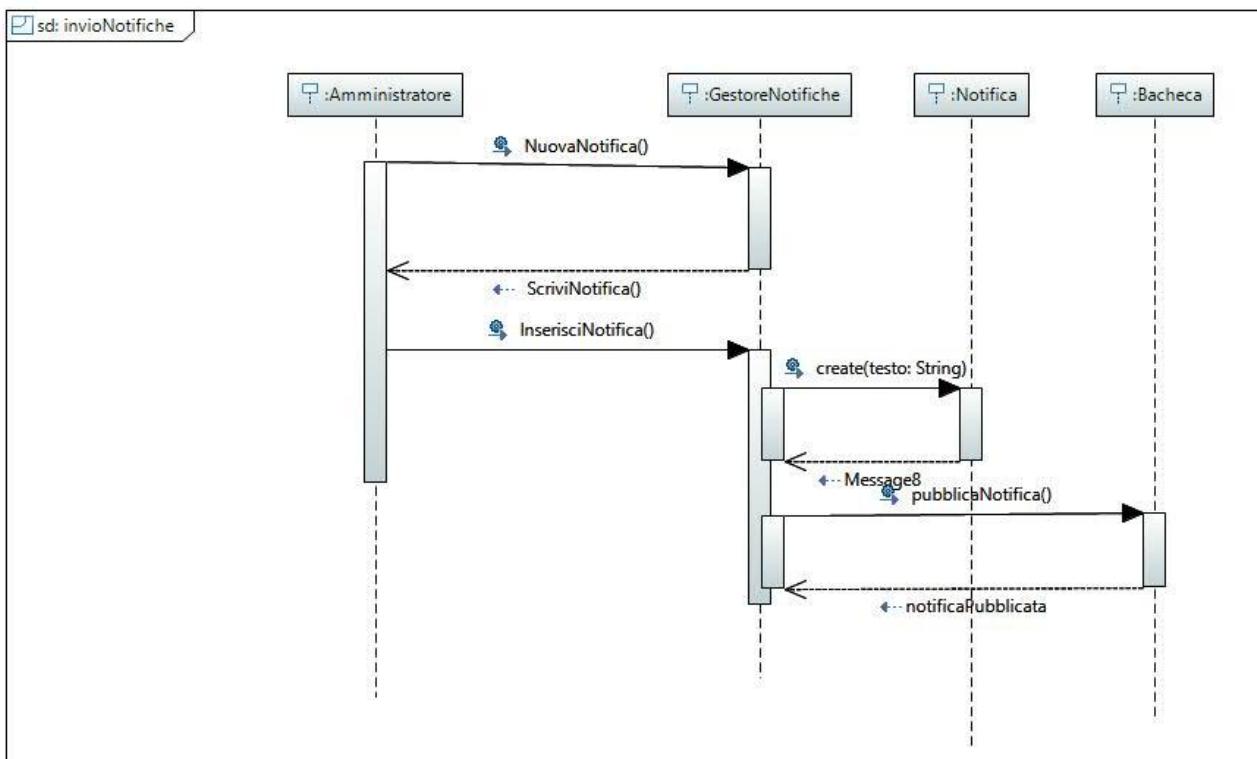
5.3.6 Sequnce Diagram – SeqVisualizzaNotifiche



Il diagramma rappresenta l'azione dell'utente di visualizzare le notifiche emesse dall'amministratore. Invia pertanto una richiesta al gestore adibito e si entrerà in una zona di "alt" in cui l'utente sceglierà di essere geolocalizzato o di scegliere manualmente la città al fine di filtrare solo le notifiche rivolte alla sua città.

5.3.7 Sequence Diagram – SeqInvioNotifiche

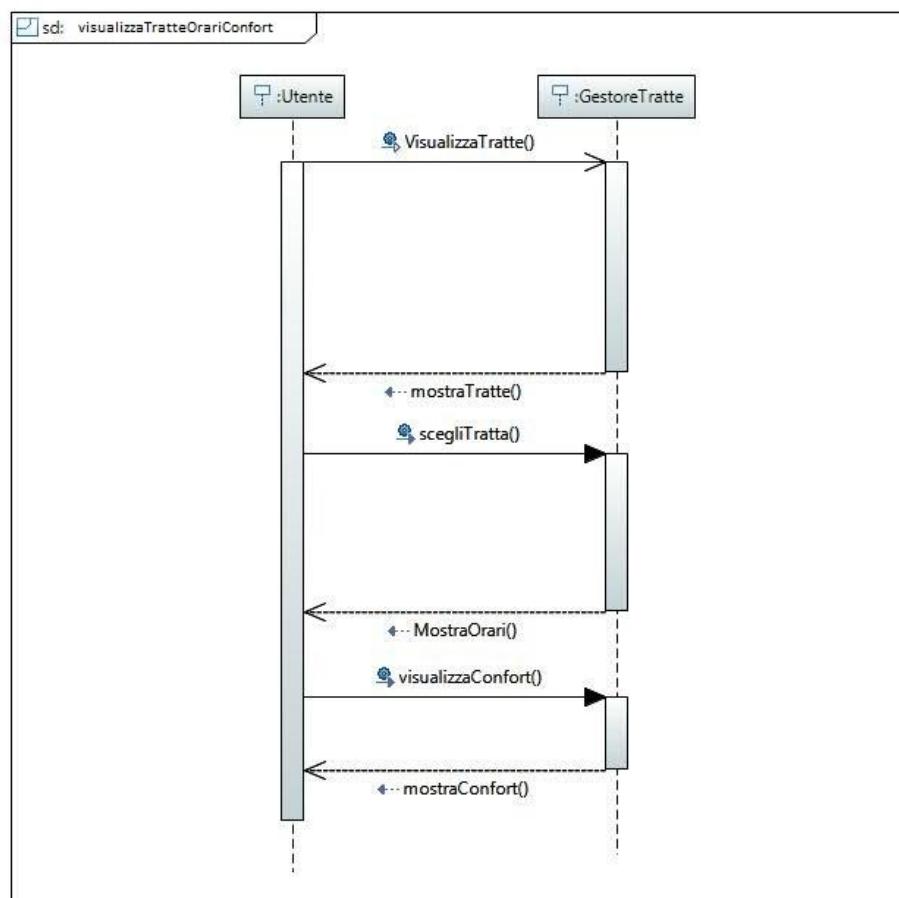
22



Tale diagramma descrive come l'amministratore possa inserire delle notifiche all'interno di una bachecca apposita. L'amministratore chiederà dapprima al gestore delle notifiche di creare una nuova notifica. Il gestore darà l'assenso e si potrà quindi scrivere il corpo della notifica. Una volta finito il gestore creerà una istanza di notifica e successivamente verrà pubblicata nella bachecca virtuale.

5.3.8 Sequence Diagram – SeqVisualizzaTratteOrariConfort

23

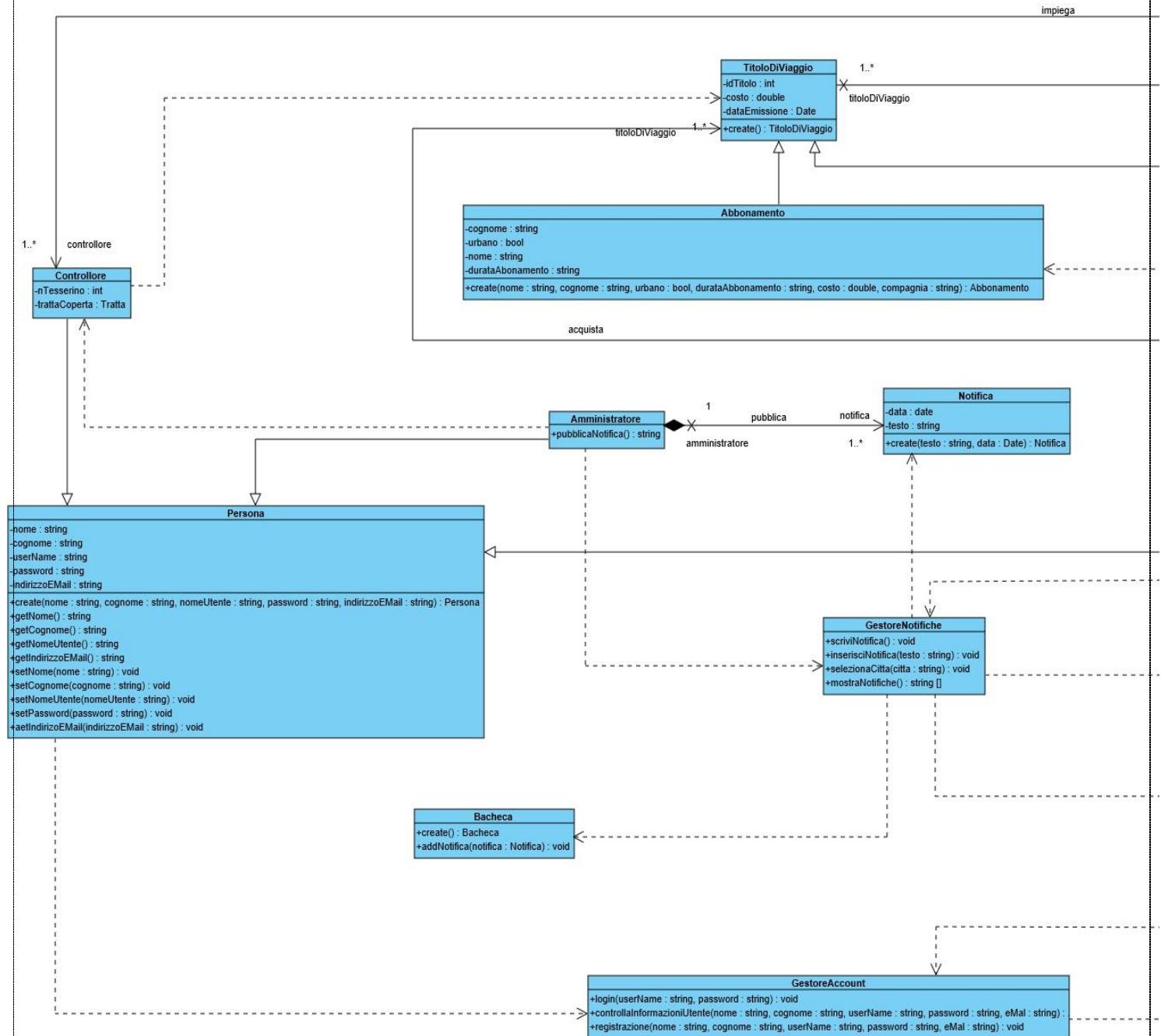


Questo diagramma mostra come l'utente visualizzi e scelga la tratta desiderata e visualizzi inoltre il relativo grado di confort.

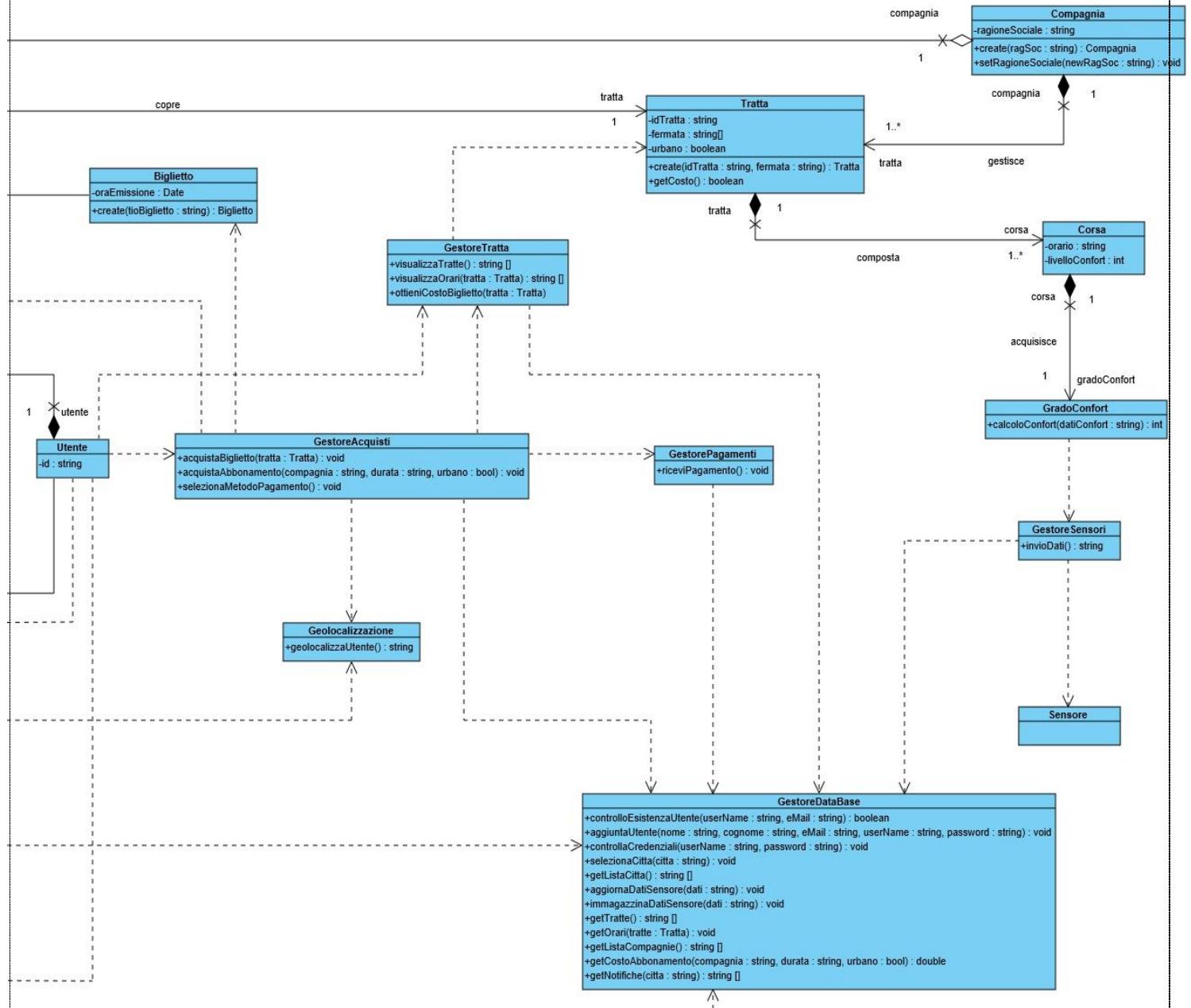
6 Modelli di progettazione

24

6.1 Class Diagram 1/2



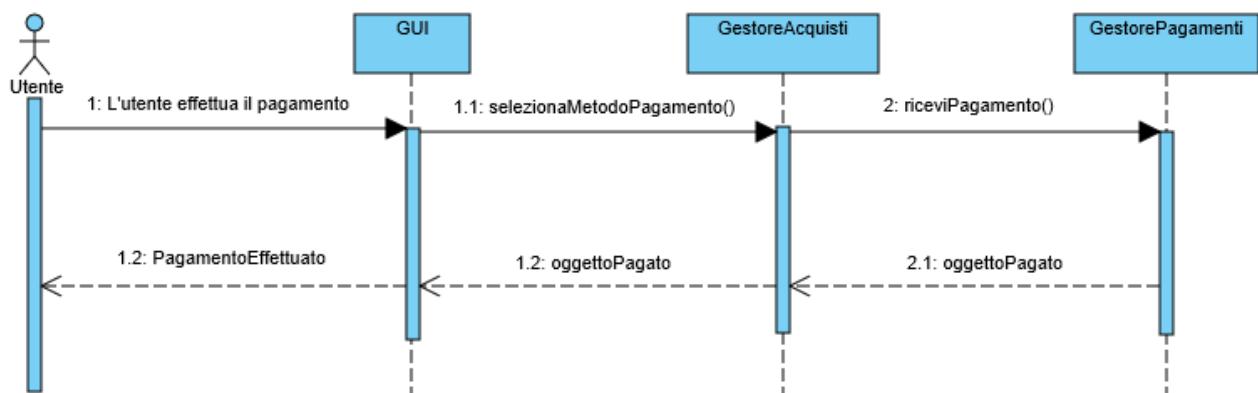
6.1 Class Diagram 2/2



Il Class Diagram di progettazione è stato modificato in determinati aspetti tecnici e concettuali:

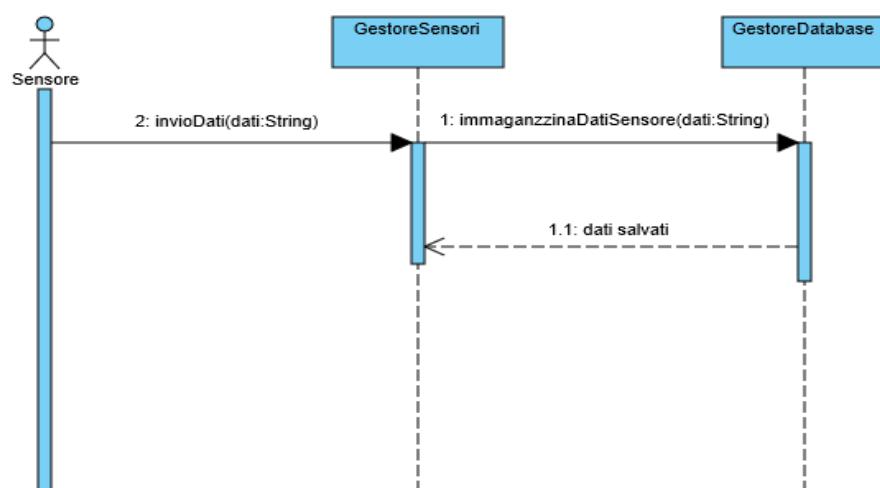
- sono state sostituite le associazioni con le più adeguate composizioni () o aggregazioni ();
- abbiamo eliminato le generalizzazioni della classe "Tratta";
- abbiamo aggiunto la macro classe "GestoreDatabase", che funge da centro di stoccaggio e modifica dei dati e la classe "GestoreTratta";
- abbiamo infine aggiunto attributi e metodi a tutte le classi e impostato loro l'opportuna visibilità.

6.2.1 Sequence Diagram – SeqPagamento



La differenza di questo diagramma di progettazione con il suo corrispettivo di analisi risiede unicamente nell'introduzione di una interfaccia grafica (GUI) a cui sono affidati i compiti di interazione con il gestore acquisti.

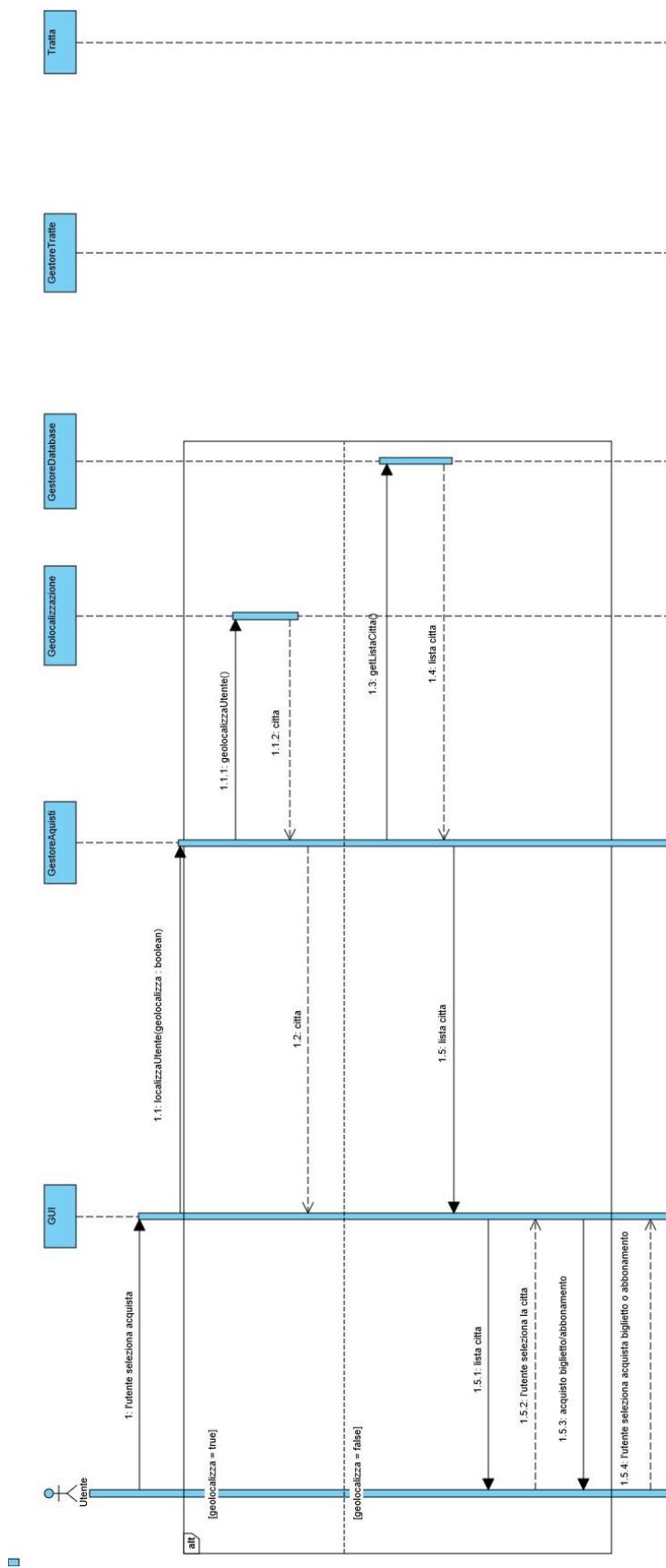
6.2.2 Sequence Diagram – SeqInvioDati



La differenza riscontrabile dall'analisi è la presenza del gestore del database di sistema che viene adibito all'immagazzinamento dei dati raccolti dal sensore.

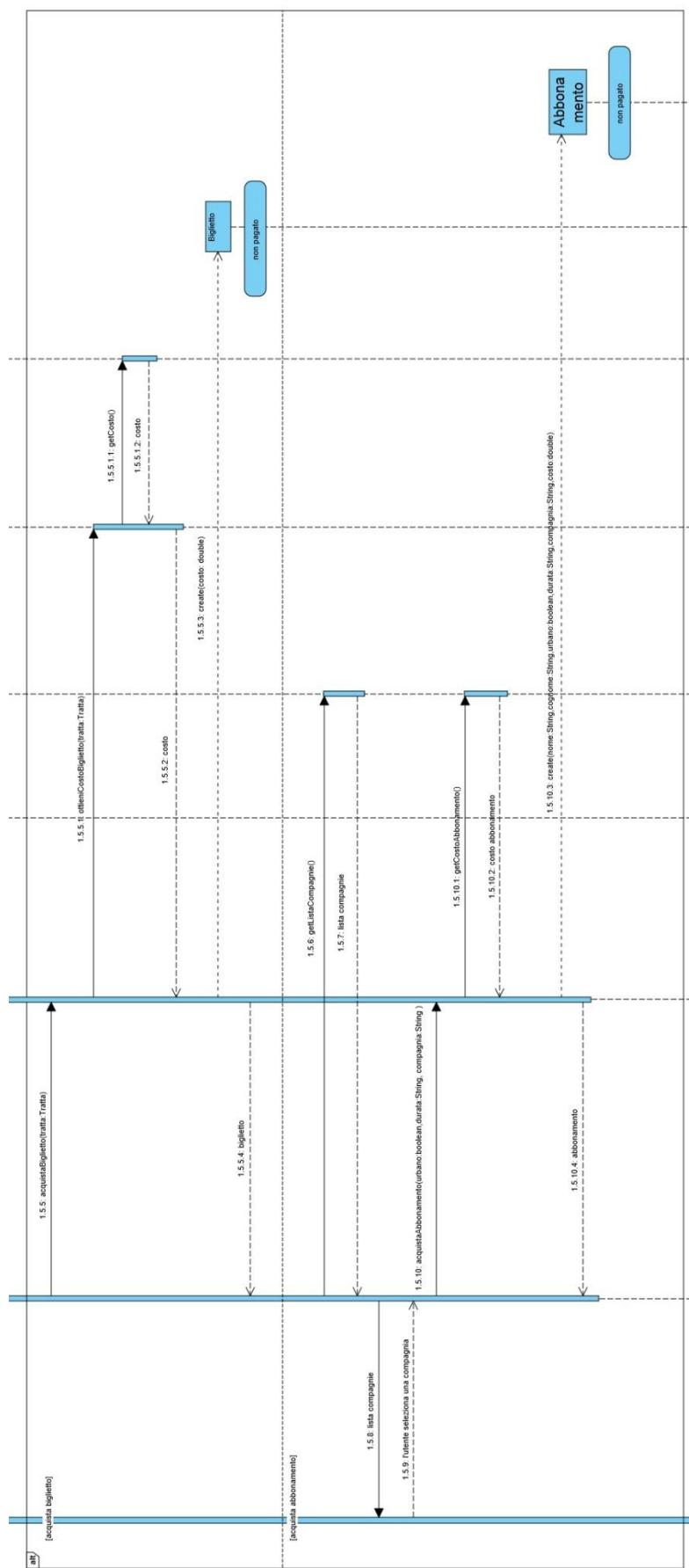
6.2.3 Sequence Diagram – SeqAcquistoBA – 1/3

27



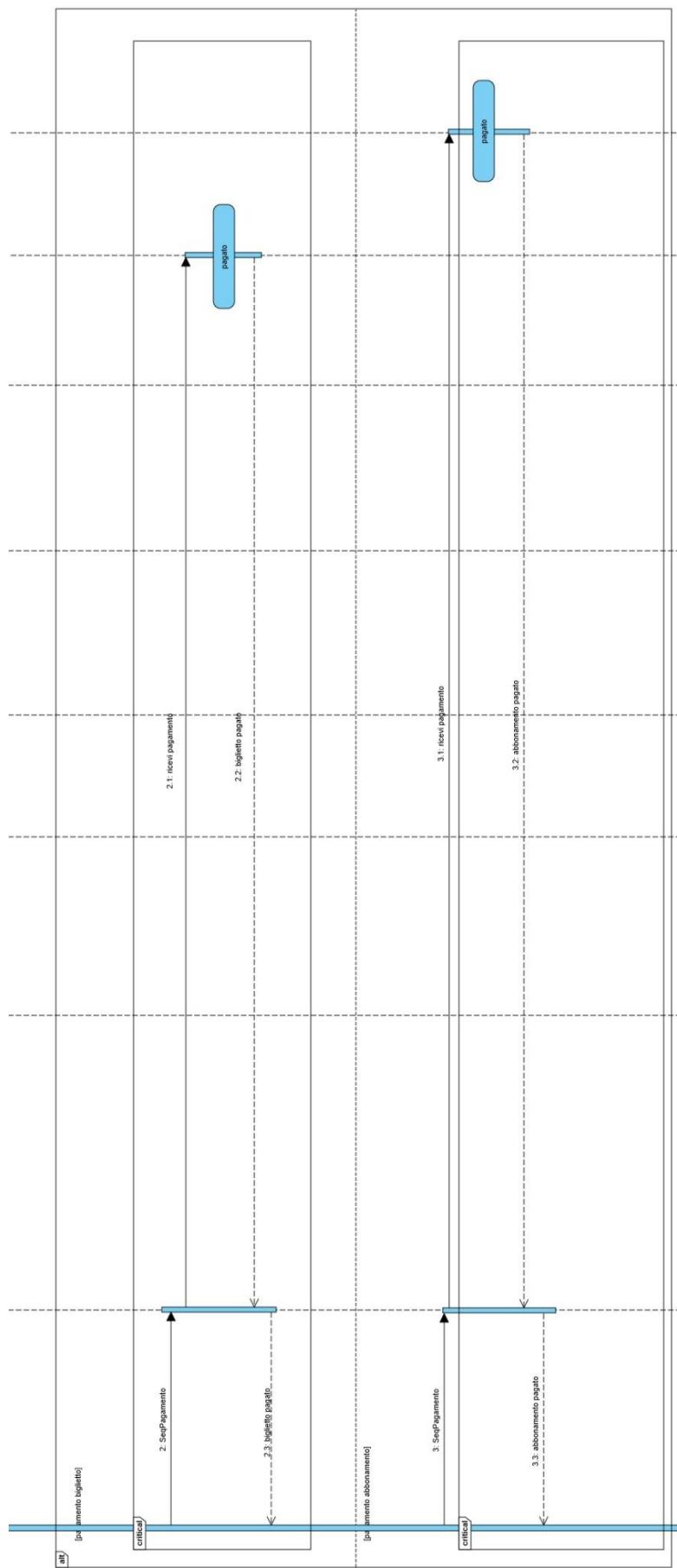
6.2.3 Sequence Diagram – SeqAcquistoBA – 2/3

28



6.2.3 Sequence Diagram – SeqAcquistoBA – 3/3

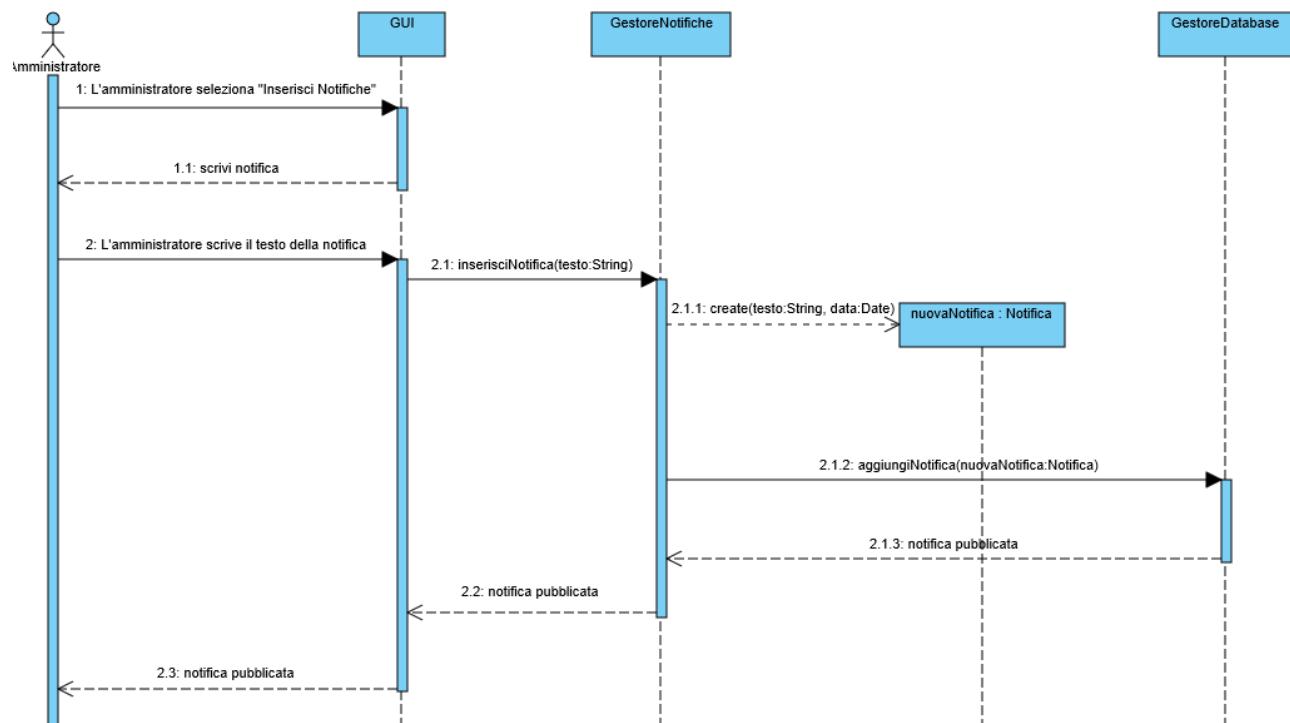
29



Le differenze riscontrate con il corrispettivo diagramma di analisi sono:

- L'introduzione di una interfaccia grafica su cui ci si appoggia per la chiamata dei vari metodi ai gestori;
- L'inserimento della classe "GestoreDatabase";
- Un più corretto uso semantico delle istanze di classe create nei giusti tempi;
- Un più corretto uso dei messaggi di ritorno;
- La scelta di migliori metodi a partire da classi di progettazione più ricche e dettagliate.

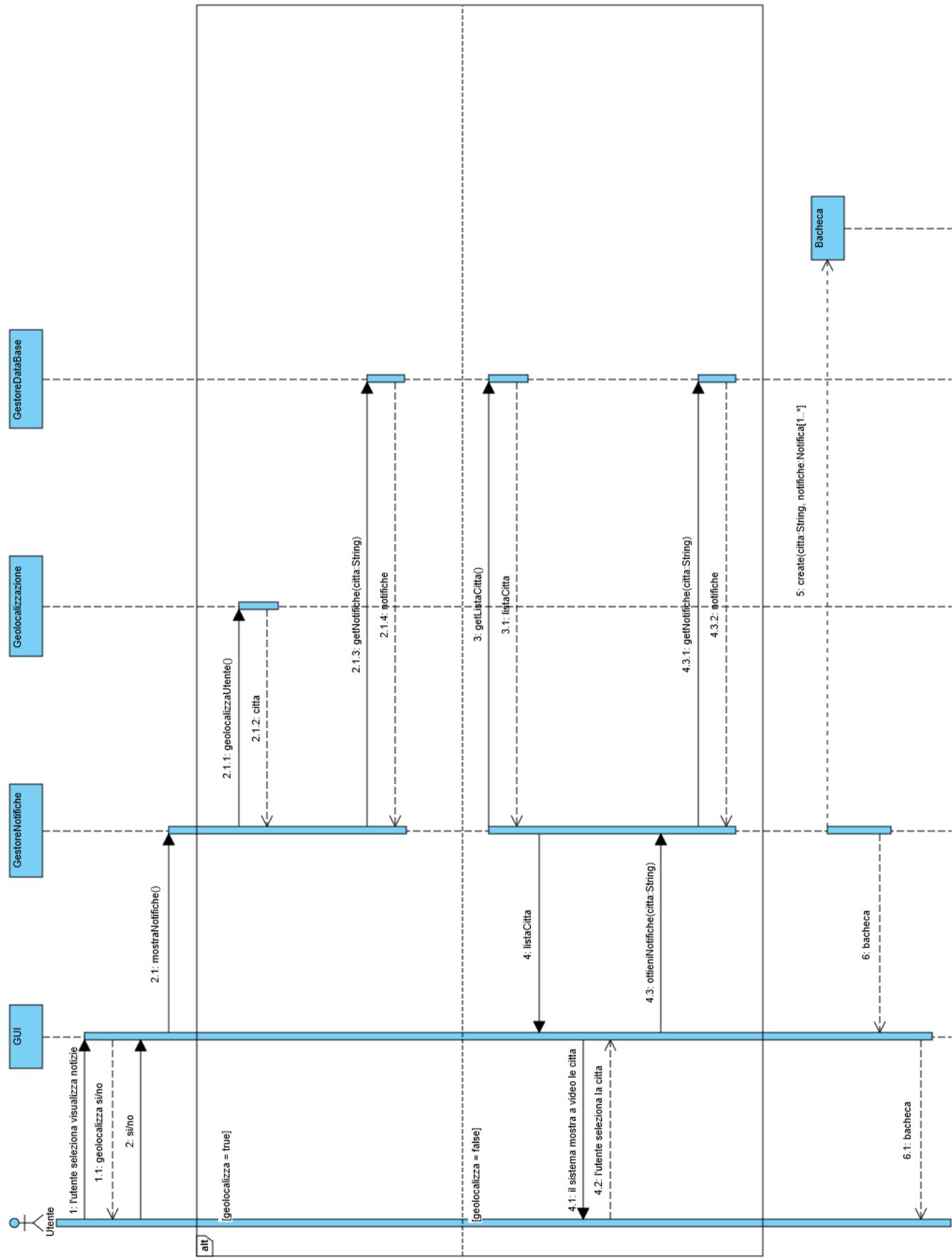
6.2.4 Sequence Diagram – SeqInvioNotifiche



La differenza di questo diagramma di progettazione con il suo corrispettivo di analisi risiede unicamente nell'introduzione di una interfaccia grafica (GUI) a cui sono affidati i compiti di interazione con il gestore notifiche.

6.2.5 Sequence Diagram – SeqVisualizzaNotifiche

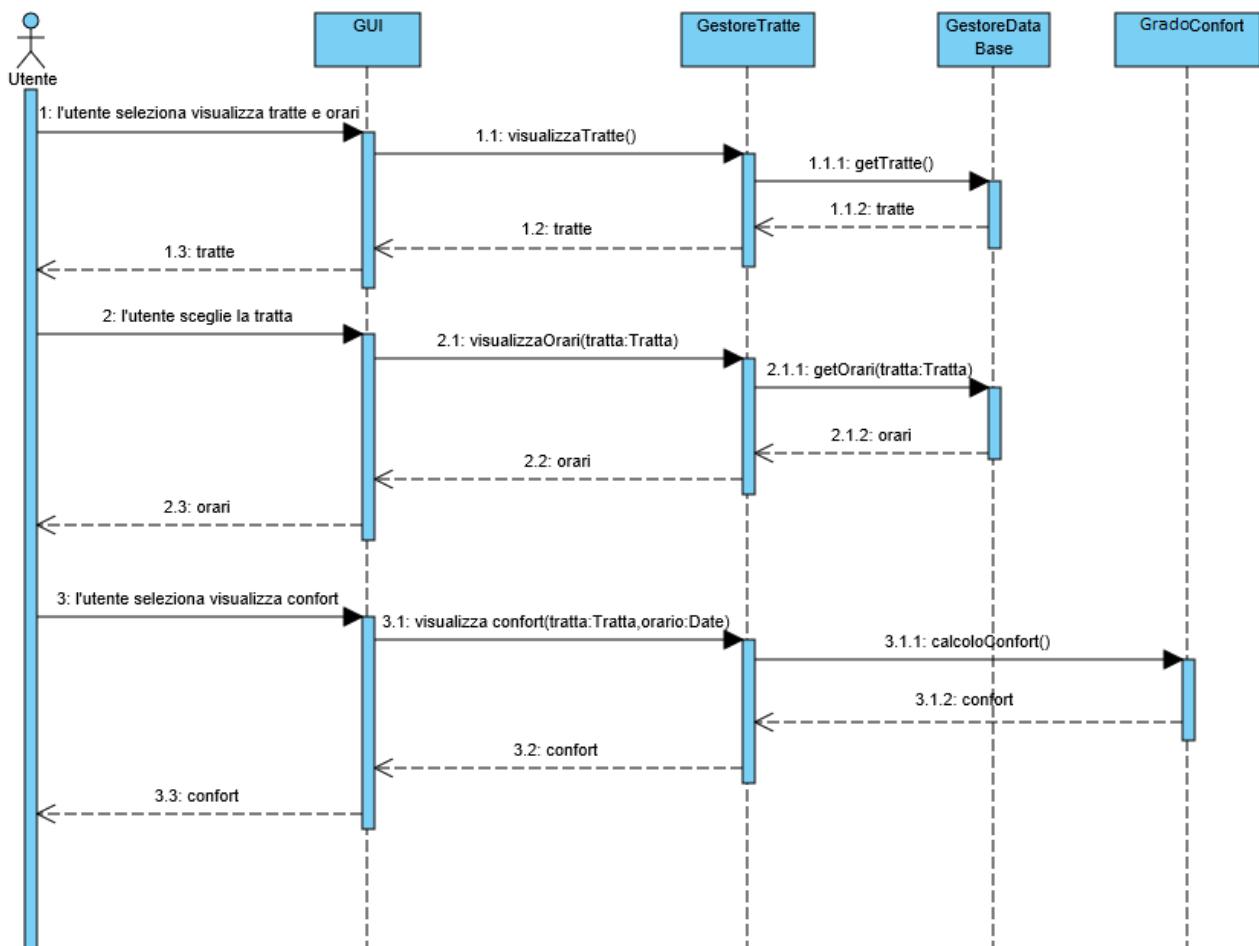
31



Le differenze riscontrate in questo diagramma risiedono:

- nell'inserimento di una interfaccia grafica (GUI) ;
- nel più corretto svolgimento delle azioni condizionali nell'area "alt";
- nell'inserimento della classe "GestoreDatabase" come pool da cui riprendere la lista delle città che adoperano il nostro servizio e le notifiche inerenti una specifica città;
- l'inserimento di una istanza della classe "Bacheca" e la sua visualizzazione all'interno della GUI.

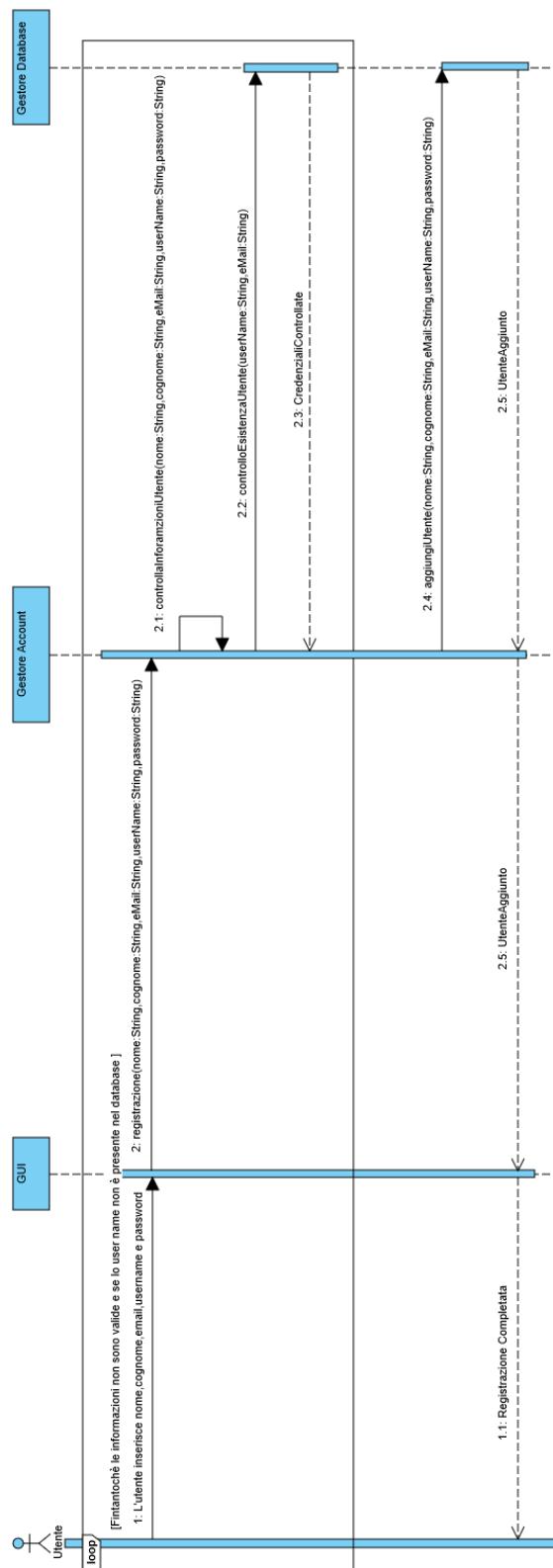
6.2.6 Sequence Diagram – SeqVisualizzaTratteOrariConfort



In fase di progettazione questo diagramma è stato ampliato aggiungendo un'interfaccia grafica e le due classi "GestoreDatabase" e "GradoConfort". Alla prima classe introdotta viene richiesto di immagazzinare la lista di tutte le tratte esistenti e dei relativi orari, mentre alla seconda di dover calcolare, tramite i dati ricevuti dai sensori su di una specifica corsa, il grado totale di confort. Tutti i messaggi di ritorno vengono visualizzati sulla GUI.

6.2.7 Sequence Diagram – SeqRegistrazione

33



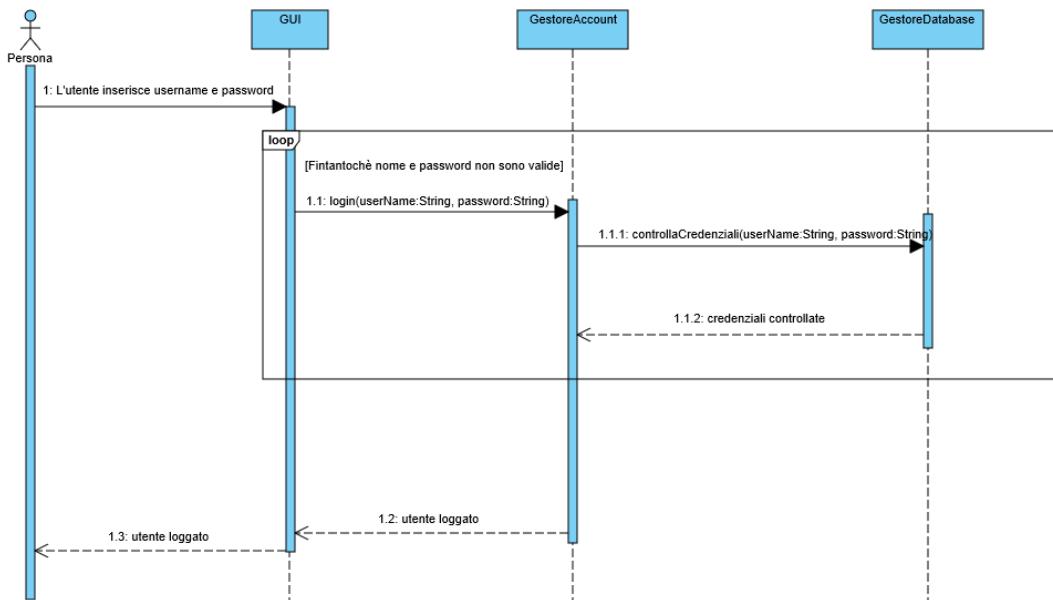
Le differenze riscontrate con l'analisi sono:

- L'uso di una classe “GestoreAccount” al posto della precedente “GestoreRegistrazioni”;
- L'introduzione di interfaccia grafica e della classe “GestoreDatabase”, nella quale immagazzinare le credenziali di accesso;

- La creazione di un record nel database per un nuovo utente anziché la creazione di una istanza a se stante di “Utente”.

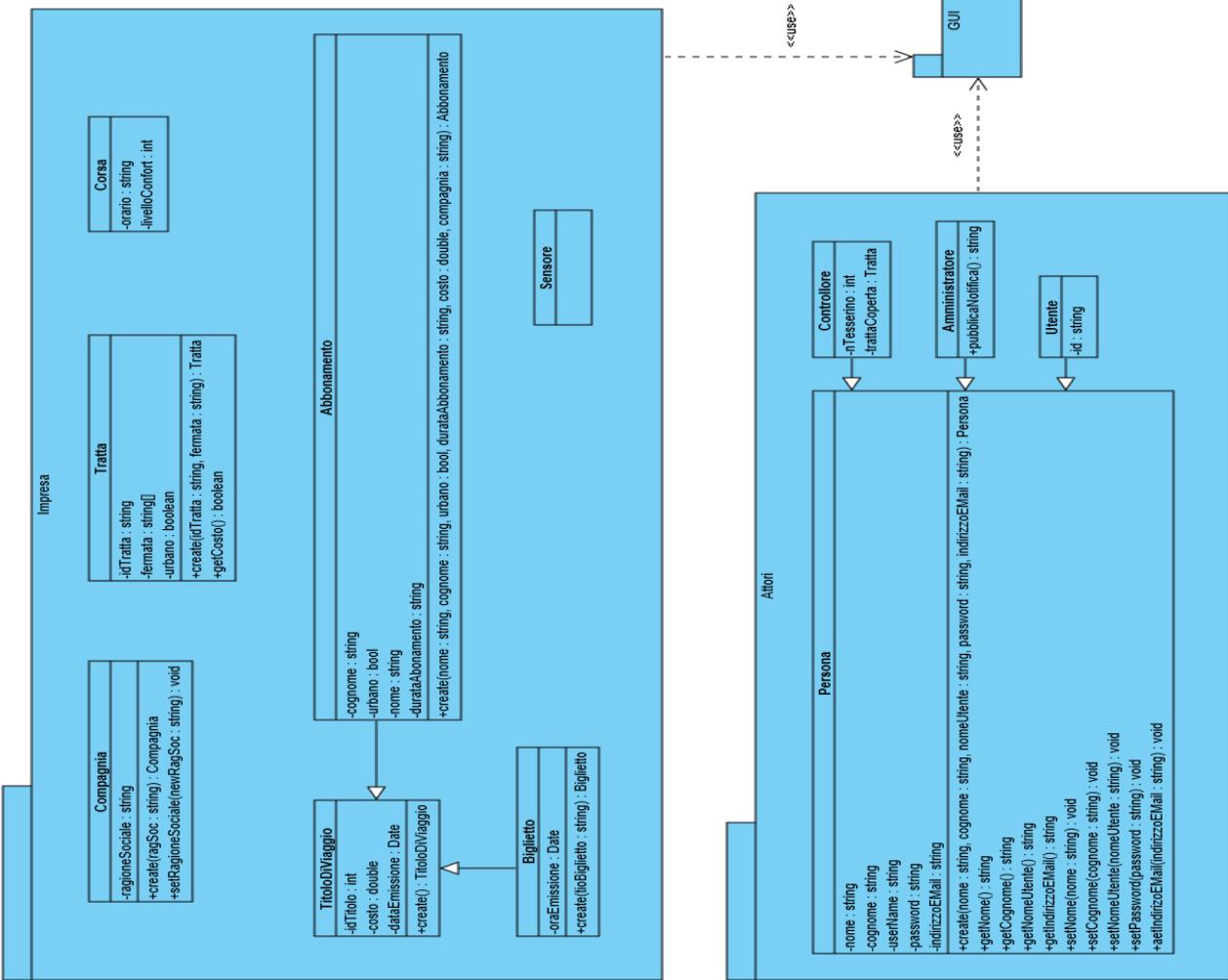
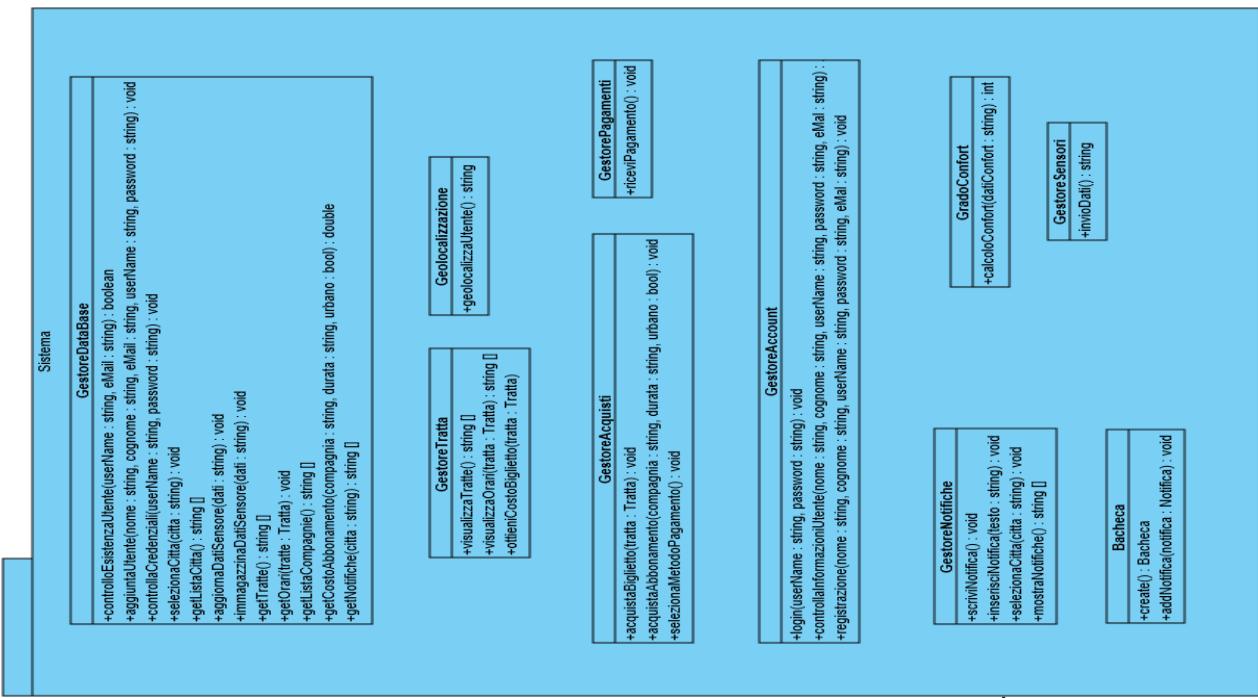
34

6.2.8 Sequence Diagram – SeqLogin



Anche in questo diagramma le uniche modifiche sono date dall'introduzione di una interfaccia grafica di gestione e dal gestore del database, a cui è affidato il compito di controllare le credenziali della persona che si vuole collegare con le credenziali che lui già possiede.

6.3 Package Diagram



Questo diagramma ci ha permesso di organizzare in modo elegante e pulito la semantica dietro al class di progettazione. Fornendo degli appositi spazi (package) siamo riusciti a creare aree semantiche, tra loro collegate, al fine di rendere più comprensibile l'entità del progetto e favorire una maggiore scalabilità e modularità in fase di prototipizzazione e sviluppo.

Abbiamo identificato tre grandi package fondamentali: "Impresa", "Attori" e "Sistema".

Il primo contiene tutte quelle classi che a noi sono parse più affini con il concetto espresso dal nome del package, quindi tutto quello che era legato all'idea della compagnia di trasporti e dei propri servizi annessi.

Il secondo contiene tutte le entità che operano nel sistema e che ne usufruiscono.

Il terzo contiene tutte quelle entità addette alla gestione delle informazioni ricavate da ogni altra parte connessa al sistema.

È stato quindi creato un package GUI dal valore semantico di mediazione. Tale package ha una dipendenza con il package "Sistema" di tipo "«accede»", ciò vuol dire che esegue una fusione privata con gli elementi pubblici della classe fornitrice. Inoltre è la classe fornitrice di tali elementi attraverso dipendenze di tipo "«usa»" con i rimanenti due package: tale scelta risiede nel fatto che la GUI possa filtrare con dei suoi metodi pubblici quelli privati ricavati da "Sistema".



Implementazione del prototipo

Nel prototipo sono state implementate le funzionalità principali che compongono il **sistema**.

Per quanto riguarda le funzionalità **utente**, sono state implementate :

1. Visualizzazione degli orari delle corse.
2. Acquisto di Biglietti e Abbonamenti.
3. Visualizzazione di notizie riguardanti disagi nel trasporto pubblico.

Per quanto riguarda invece il lato **amministratore**, è stata implementata un'unica funzionalità:

1. Pubblicazione delle notizie.

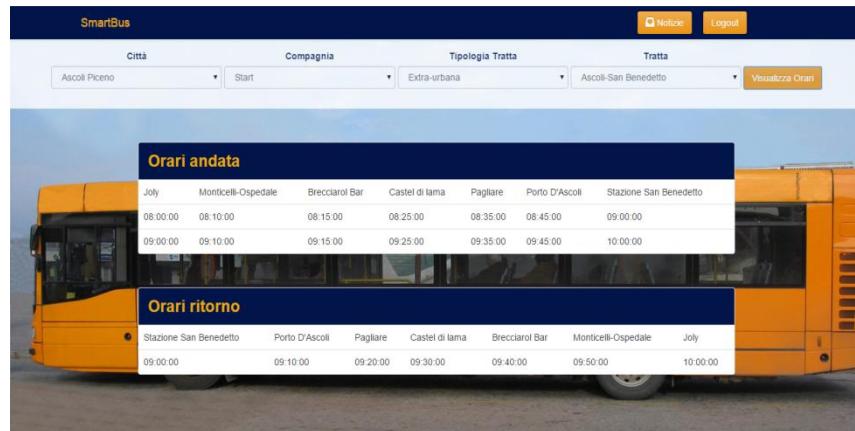
Il sito è composto da una pagina iniziale, nella quale l'utente e/o amministratore può **accedere** previa inserimento di email e password, oppure può **registrarsi** inserendo alcune generalità. Nella medesima, è inoltre presente una sezione nella quale viene fornita una **descrizione** del sito.

Una volta effettuato l'accesso come utente, si viene reindirizzati nella pagina principale del sito, nella quale in evidenza sono poste le funzionalità di **acquisto di biglietti e abbonamenti** e la **visualizzazione degli orari**. In alto nella barra di navigazione troviamo il pulsante **Notizie**, attraverso il quale è possibile visitare la pagina nella quale sono pubblicate le info riguardanti disagi nel servizio di trasporto, e il pulsante di **Logout**, attraverso il quale si esce dalla propria area riservata.



Visualizza Tratte e Orari

In questa sezione del sito è possibile consultare, in base alla scelta della città, compagnia, tipo di tratta e tratta, gli **orari** degli autobus lungo tutto il percorso previsto dalla tratta selezionata. Il sito, mette a disposizione sia gli orari di andata che di ritorno.



Visualizza Notizie

Nella pagina di **visualizzazione delle notizie**, è possibile consultare gli aggiornamenti riguardanti disagi e/o disservizi nei trasporti pubblici. In questa pagina compaiono le notizie pubblicate dagli amministratori del sito i quali provvedono a informare gli utenti del trasporto pubblico nel caso in cui non sia possibile fornire loro il servizio normalmente previsto.



Acquisto Biglietti e Abbonamenti

All'interno della pagina di acquisto biglietti e abbonamenti, l'utente può in maniera molto semplice e veloce, acquistare un biglietto o un abbonamento. Dopo aver scelto citta compagnia e tipo di tratta, vengono richiesti all'utente alcuni dati personali nel caso stia acquistando un abbonamento. Poi in entrambi i casi verranno richiesti i dati della carta

attraverso la quale si intende effettuare il pagamento. Infine verrà generato in automatico il pdf, che rappresenta il biglietto o abbonamento acquistato.

39



Pagina Amministratore

Se le credenziali inserite nei campi di login corrispondono a quelle di un amministratore, si accede alla pagina amministratore, nella quale, è possibile pubblicare le notizie riguardanti gli aggiornamenti in tempo reale di disagi nel trasporto pubblico.



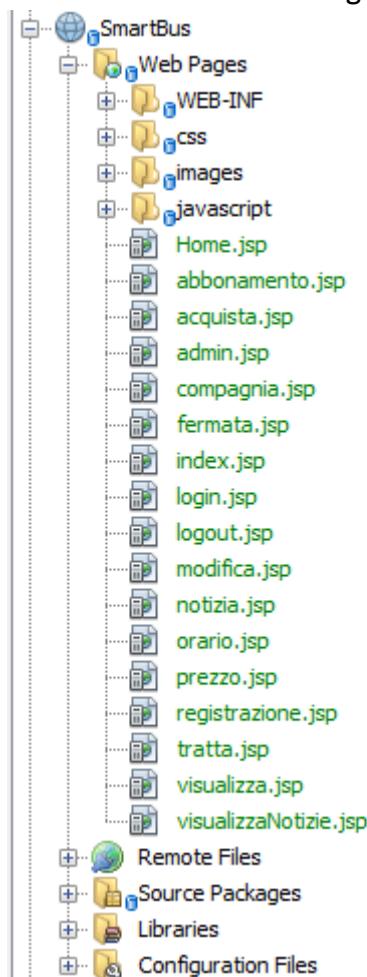
Tecnologie utilizzate

Il prototipo è stato sviluppato come sito web, utilizzando come tecnologie **HTML5**, **CSS** e **JavaScript** per quanto riguarda il **lato client**, e **JSP** (JavaServer Pages) per il **lato server**. Nello sviluppo del lato client è inoltre stata utilizzata **Bootstrap**.

Bootstrap è una raccolta di strumenti liberi per la creazione di siti e applicazioni per il Web. Essa contiene modelli di progettazione basati su HTML e CSS, sia per la tipografia, che per le varie componenti dell'interfaccia, come moduli, buttoni e navigazione, e altri componenti dell'interfaccia, così come alcune estensioni opzionali di JavaScript. Per il database è stato utilizzato un **DB relazionale**, composto da uno standalone attraverso il quale ci si interfaccia con **HeidisSQL**.

Struttura del percorso dei file

L'applicazione è stata strutturata nella seguente maniera:



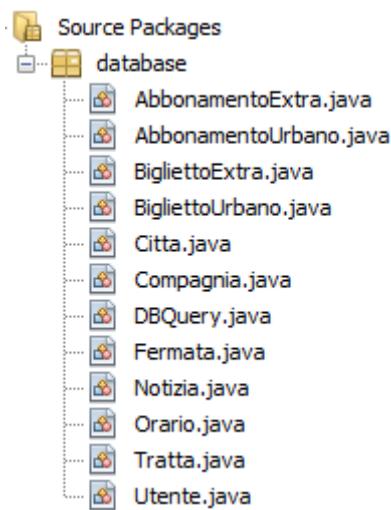
L'applicazione è stata suddivisa in quattro sotto cartelle principali:

1. **Web-Pages** : contiene tutte le pagine web, i file javascript, i fogli di stile e le immagini utilizzate all'interno del sito;
2. **Source Packages** : al suo interno sono presenti tutte le classi java che rappresentano gli oggetti presenti sul database e la classe DBQuery che ci permette di interagire col database effettuando delle query;
3. **Libraries** : contenente le librerie importate (come ad esempio mysql-connector-java e json-simple) oltre che alle altre librerie già presenti di default;
4. **Configuration Files** : cartella che contiene il file di configurazione web.xml nel quale si trovano i parametri che formano la stringa di connessione.

Classi Java

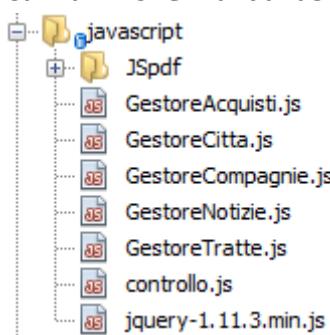
All'interno della sottocartella **Source Packages** troviamo il package **database**, il quale contiene le classi Java. Queste sono le classi **DBQuery**, la quale ha tutti i metodi che operano query e update sul database, e tutte le altre classi che rispecchiano le tabelle del db, quali Utente, Notizia, Orario, Tratta etc...

Tali classi vengono sfruttate quando vengono effettuate delle query nel database.



Javascript

La cartella JavaScript contiene tutti i file di estensione **.js**; troviamo al suo interno innanzitutto i file **Gestori**, i quali rispecchiano il più possibile quelli presenti nel Class Diagram. Vi è poi la cartella **JSpdf**, la quale contiene una libreria JavaScript, sfruttata per creare file formato pdf per la creazione di biglietti e abbonamenti. Infine come si può notare, è stata importata la libreria JavaScript **JQuery**.



Struttura del database

Il database utilizzato, è un database relazionale MySQL composto da uno standalone, e da l'interfaccia grafica HeidiSQL.

Le tabelle create sono quelle mostrate in figura:

smartbus	432,0 KB
abbonamento_ext...	48,0 KB
abbonamento_ur...	48,0 KB
biglietto_extra	48,0 KB
biglietto_urbano	48,0 KB
citta	16,0 KB
compagnia	16,0 KB
corsa	32,0 KB
fermata	32,0 KB
notizie	16,0 KB
orario	48,0 KB
tratta	64,0 KB
utente	16,0 KB

Le tabelle sono strutturate in maniera tale da mantenere i **vincoli di integrità**, attraverso la creazione di **chiavi esterne**. Sono state collegate fra di loro in maniera tale da rispettare il più possibile la struttura del Class Diagram.

Tabelle

Utente

Tabella nella quale sono presenti le generalità degli utenti registrati al sito web:

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default	Comment	Collation
1	ID	INT	10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREMENT		
2	Ruolo	VARCHAR	50		<input checked="" type="checkbox"/>		NULL		
3	Cancellato	INT	1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		
4	Nome	VARCHAR	50		<input checked="" type="checkbox"/>		NULL		
5	Cognome	VARCHAR	50		<input checked="" type="checkbox"/>		NULL		
6	Email	VARCHAR	50		<input checked="" type="checkbox"/>		NULL		
7	Password	VARCHAR	50		<input checked="" type="checkbox"/>		NULL		
8	Sesso	VARCHAR	1		<input checked="" type="checkbox"/>		NULL		
9	Residenza	VARCHAR	50		<input checked="" type="checkbox"/>		NULL		
10	Data_nascita	VARCHAR	50		<input checked="" type="checkbox"/>		NULL		
11	Luogo_nascita	VARCHAR	50		<input checked="" type="checkbox"/>		NULL		

Tratta

Tabella in cui compaiono i nomi delle tratte, il tipo delle tratte (urbana o extra-urbana), il riferimento alla città, alla fermata di arrivo e di partenza:

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default	Comment	Collation
1	ID	INT	10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREMENT		
2	Nome_tratta	VARCHAR	50	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		
3	Compagnia	INT	50	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		
4	Tipo	VARCHAR	50	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		
5	Partenza	INT	10	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		
6	Arrivo	INT	10	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		

Corsa

Tabella in cui sono presenti gli orari di partenza delle corse, il tipo delle corse (ovvero se la corsa è di andata o di ritorno) e il riferimento alla tratta:

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default	Comment	Collation
1	ID	INT	10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREMENT		
2	Orario_partenza	TIME		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default		
3	AR	INT	1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default		
4	Tratta	INT	10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default		

Orario

Tabella in cui compaiono gli orari delle varie corse con il riferimento alla relativa fermata:

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default	Comment	Collation
1	ID	INT	10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREMENT		
2	Fermata	INT	10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default		
3	Orario	TIME		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default		
4	Corsa	INT	10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default		

Fermata

Tabella in cui sono presenti i nomi delle fermate con il riferimento alla città in cui si trovano:

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default	Comment	Collation
1	ID	INT	10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREMENT		
2	Nome_fermata	VARCHAR	50			<input type="checkbox"/>	No default		
3	Citta	INT	10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default		

Compagnia

Tabella in cui compaiono i nomi delle diverse compagnie:

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default	Comment	Collation
1	ID	INT	10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREMENT		
2	Nome_compagnia	VARCHAR	50		<input checked="" type="checkbox"/>		NULL		

Città

Tabella in cui compaiono il nome delle città:

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default	Comment	Collation
1	ID	INT	10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREMENT		
2	Nome_citta	VARCHAR	50		<input checked="" type="checkbox"/>		NULL		

Notizia

Tabella in cui sono contenute le notizie pubblicate dall'amministratore:

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default	Comment	Collation
1	ID	INT	10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREMENT		
2	Notizia	VARCHAR	50		<input checked="" type="checkbox"/>		NULL		
3	Data	VARCHAR	50		<input checked="" type="checkbox"/>		NULL		
4	Orario	VARCHAR	50		<input type="checkbox"/>		No default		

Biglietti

Tabelle in cui sono indicati i prezzi dei due tipi di biglietto:

Biglietto extra-urbano

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default	Comment	Collation
1	ID	INT	10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREMENT		
2	Prezzo	FLOAT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		
3	Compagnia	INT	10	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		
4	Tratta	INT	10	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		

Biglietto urbano

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default	Comment	Collation
1	ID	INT	10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREMENT		
2	Prezzo	FLOAT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		
3	Compagnia	INT	10	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		
4	Tratta	INT	10	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		

Abbonamenti

Tabelle in cui sono indicati i prezzi dei due tipi di abbonamento:

Abbonamento urbano

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default	Comment	Collation
1	ID	INT	10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREMENT		
2	Prezzo	FLOAT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		
3	Compagnia	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		
4	Citta	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		

Abbonamento extra-urbano

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default	Comment	Collation
1	ID	INT	10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREMENT		
2	Prezzo	FLOAT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		
3	Compagnia	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		
4	Tratta	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		

Query

Di seguito viene mostrato un esempio di query al db:

```

public static ArrayList<Compagnia> getCompagnia(int citta,ServletContext cont){
    ArrayList<Compagnia> accompagnia=new ArrayList();
    try{
        Class.forName("com.mysql.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://" + cont.getInitParameter("ip") + "/" + cont.getInitParameter("database") + "?" +
        "user=" + cont.getInitParameter("user") + "&password=" + cont.getInitParameter("dbpassword"));
        PreparedStatement pstmt = con.prepareStatement(" SELECT distinct c.ID, c.Nome_compagnia, f.Citta as citta " +
                " FROM compagnia as c " +
                " join tratta as t on c.ID=t.Compagnia " +
                " join corsa as co on co.Tratta=t.ID " +
                " join orario as o on o.Corsa=co.ID " +
                " join fermata as f on f.ID=o.Fermata " +
                " where citta like ? ");
        pstmt.setInt(1, citta);
        ResultSet rs = pstmt.executeQuery();
        while (rs.next()){
            int id=rs.getInt("ID");
            String nome_compagnia=rs.getString("Nome_compagnia");
            int id_citta=rs.getInt("citta");
            Compagnia c= new Compagnia(id, nome_compagnia, id_citta);
            accompagnia.add(c);
        }
        con.close();
    }
    catch (Exception e){
        System.out.println("Errore con DB o Query errata");
        e.printStackTrace();
    }
    return accompagnia;
} // End getCompagnia
}

```

E' stata scelta questa **query** per la sua complessità, creatasi per via della composizione del **class diagram**.

Il **metodo** sopra mostrato è quello utilizzato per effettuare la query che estrapola dal database le compagnie per una data città; tuttavia non vi è un collegamento diretto fra le tabelle “Compagnia” e “Citta”, quindi è stato necessario eseguire una serie di **join** fra diverse tabelle fino ad ottenere il risultato desiderato.

La struttura generale dei metodi che effettuano operazioni **CRUD** sul database è sempre la stessa. Nelle prime righe del metodo viene effettuata la **connessione al database**; viene poi descritta come stringa l'operazione che verrà eseguita sulla base di dati.

Il db restituisce, nel caso di una query, una stringa per ogni riga estratta. Con un **ciclo while** si va a scorrere ogni riga estratta e, in questo caso, viene creato un oggetto “Compagnia” che poi viene aggiunto all' **ArrayList** di tipo “Compagnia” che viene restituito dal metodo. Tutta l'operazione è posta all'interno di un blocco “**try-catch**”, al fine di catturare le eccezioni generate dal codice del metodo.