PowerEnjoy

Riccardo Redaelli
Nicola Sosio
Maria Chiara Zaccardi

Politecnico di Milano

6 Marzo 2017

Introduzione

Introduzione

PowerEnjoy è una società di car sharing che offre il solo utilizzo di macchine elettriche.
Il sistema che andremo a sviluppare deve permettere agli utenti di noleggiare una macchina per brevi viaggi e di inviare notifiche agli operatori con le richieste di riparazione o di assistenza.
PowerEnjoy vuole incentivare il comportamente virtuoso degli utenti attraverso sconti.

Analisi dei requisiti

Attori

- **Visitor**: persone che devono registrarsi per usufruire del servizio. Possono visualizzare i form per accedere o registrarsi e una breve descrizione del servizio offerto.
- **User**: utenti che, dopo aver eseguito il login, sono abilitate ai servizi offerti.
- **Operator**: dipendenti della società che, dopo il login, possono visualizzare la macchina sulla quale devono effettuare l'operazione.
- **PayPal**: è un attore passivo, utilizzato dal sistema per il pagamento dei noleggi.

Attori

- **Visitor**: persone che devono registrarsi per usufruire del servizio. Possono visualizzare i form per accedere o registrarsi e una breve descrizione del servizio offerto.
- **User**: utenti che, dopo aver eseguito il login, sono abilitate ai servizi offerti.
- **Operator**: dipendenti della società che, dopo il login, possono visualizzare la macchina sulla quale devono effettuare l'operazione.
- **PayPal**: è un attore passivo, utilizzato dal sistema per il pagamento dei noleggi.

Attori

- **Visitor**: persone che devono registrarsi per usufruire del servizio. Possono visualizzare i form per accedere o registrarsi e una breve descrizione del servizio offerto.
- **User**: utenti che, dopo aver eseguito il login, sono abilitate ai servizi offerti.
- **Operator**: dipendenti della società che, dopo il login, possono visualizzare la macchina sulla quale devono effettuare l'operazione.
- **PayPal**: è un attore passivo, utilizzato dal sistema per il pagamento dei noleggi.

Attori

- **Visitor**: persone che devono registrarsi per usufruire del servizio. Possono visualizzare i form per accedere o registrarsi e una breve descrizione del servizio offerto.
- **User**: utenti che, dopo aver eseguito il login, sono abilitate ai servizi offerti.
- **Operator**: dipendenti della società che, dopo il login, possono visualizzare la macchina sulla quale devono effettuare l'operazione.
- **PayPal**: è un attore passivo, utilizzato dal sistema per il pagamento dei noleggi.

Assunzioni 1/2

- Gli operatori ricevono le credenziali di accesso al momento del loro conferimento d'incarico.
- Nelle ore di servizio gli operatori sono sempre disponibili e connessi all'applicazione.
- Le operazioni vengono sempre assegnate all'operatore più vicino alla macchina.
- Per sbloccare la macchina, una volta prenotata, è necessario inserire il codice della macchina nell'applicazione.

Assunzioni 1/2

- Gli operatori ricevono le credenziali di accesso al momento del loro conferimento d'incarico.
- Nelle ore di servizio gli operatori sono sempre disponibili e connessi all'applicazione.
- Le operazioni vengono sempre assegnate all'operatore più vicino alla macchina.
- Per sbloccare la macchina, una volta prenotata, è necessario inserire il codice della macchina nell'applicazione.

Assunzioni 1/2

- Gli operatori ricevono le credenziali di accesso al momento del loro conferimento d'incarico.
- Nelle ore di servizio gli operatori sono sempre disponibili e connessi all'applicazione.
- Le operazioni vengono sempre assegnate all'operatore più vicino alla macchina.
- Per sbloccare la macchina, una volta prenotata, è necessario inserire il codice della macchina nell'applicazione.

Assunzioni 1/2

- Gli operatori ricevono le credenziali di accesso al momento del loro conferimento d'incarico.
- Nelle ore di servizio gli operatori sono sempre disponibili e connessi all'applicazione.
- Le operazioni vengono sempre assegnate all'operatore più vicino alla macchina.
- Per sbloccare la macchina, una volta prenotata, è necessario inserire il codice della macchina nell'applicazione.

Assunzioni 2/2

- L'account viene bloccato se l'utente effettua un noleggio per cui non si verifica disponibilità di pagamento.
- Il parcheggio è consentito solo all'interno di una Safe Area, fuori da essa il noleggio non si interrompe.
- Se un noleggio termina a più di 3km da una power grid station, un operatore viene incaricato di spostare la macchina entro i 3km.
- Una macchina in carica ricompare fra le macchine disponibili solo a carica completa.

Assunzioni 2/2

- L'account viene bloccato se l'utente effettua un noleggio per cui non si verifica disponibilità di pagamento.
- Il parcheggio è consentito solo all'interno di una Safe Area, fuori da essa il noleggio non si interrompe.
- Se un noleggio termina a più di 3km da una power grid station, un operatore viene incaricato di spostare la macchina entro i 3km.
- Una macchina in carica ricompare fra le macchine disponibili solo a carica completa.

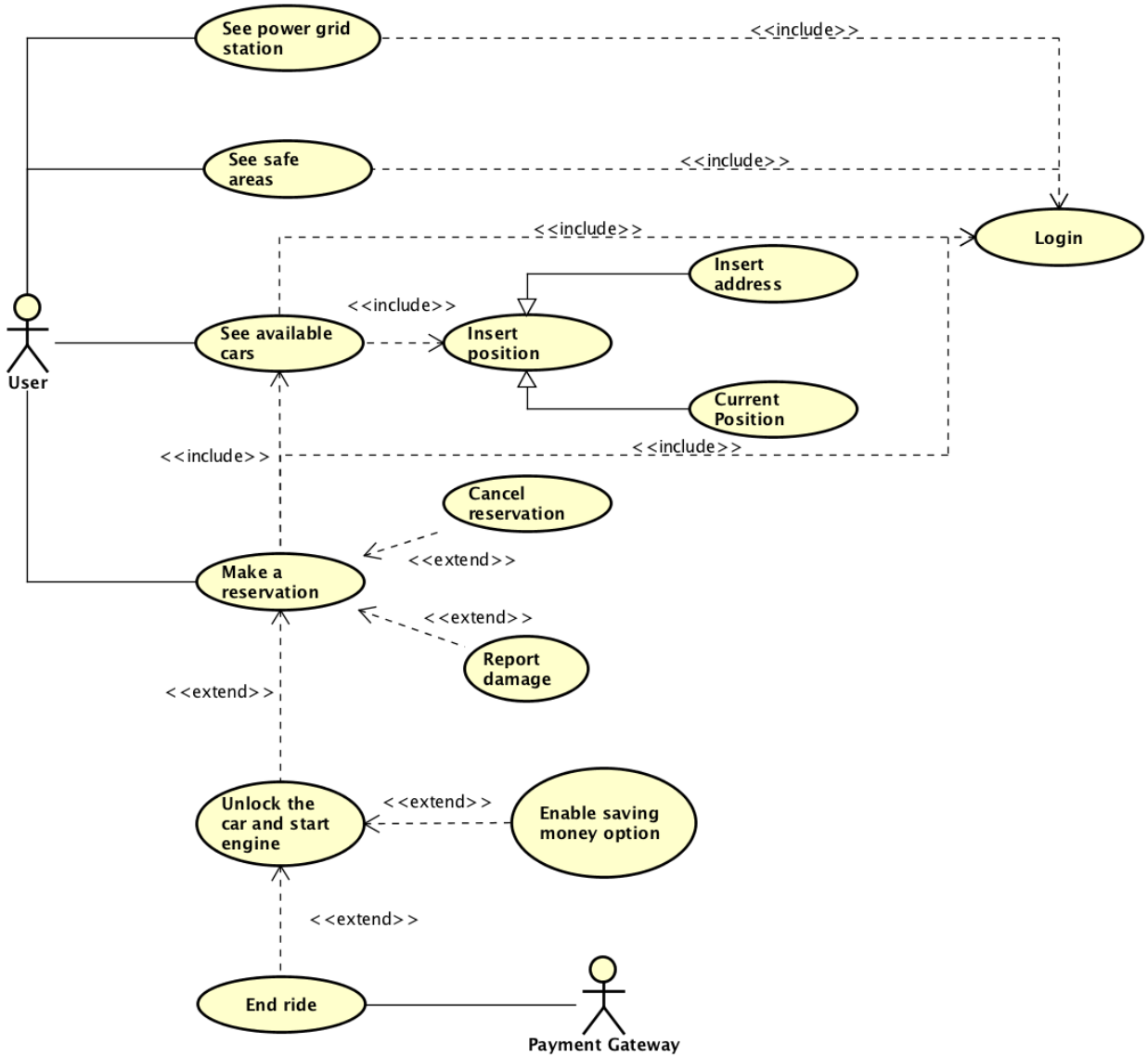
Assunzioni 2/2

- L'account viene bloccato se l'utente effettua un noleggio per cui non si verifica disponibilità di pagamento.
- Il parcheggio è consentito solo all'interno di una Safe Area, fuori da essa il noleggio non si interrompe.
- Se un noleggio termina a più di 3km da una power grid station, un operatore viene incaricato di spostare la macchina entro i 3km.
- Una macchina in carica ricompare fra le macchine disponibili solo a carica completa.

Assunzioni 2/2

- L'account viene bloccato se l'utente effettua un noleggio per cui non si verifica disponibilità di pagamento.
- Il parcheggio è consentito solo all'interno di una Safe Area, fuori da essa il noleggio non si interrompe.
- Se un noleggio termina a più di 3km da una power grid station, un operatore viene incaricato di spostare la macchina entro i 3km.
- Una macchina in carica ricompare fra le macchine disponibili solo a carica completa.

Use case 1/2


See power grid station
<<include>>
See safe areas
<<include>>
Login
<<include>>
Insert address
<<include>>
See available cars
Insert position
Current Position
User
<<include>>
<<include>>
Cancel reservation
<<extend>>
Make a reservation
<<extend>>
Report damage
<<extend>>
Unlock the car and start engine
<<extend>>
Enable saving money option
<<extend>>
End ride
Payment Gateway

Use case 2/2

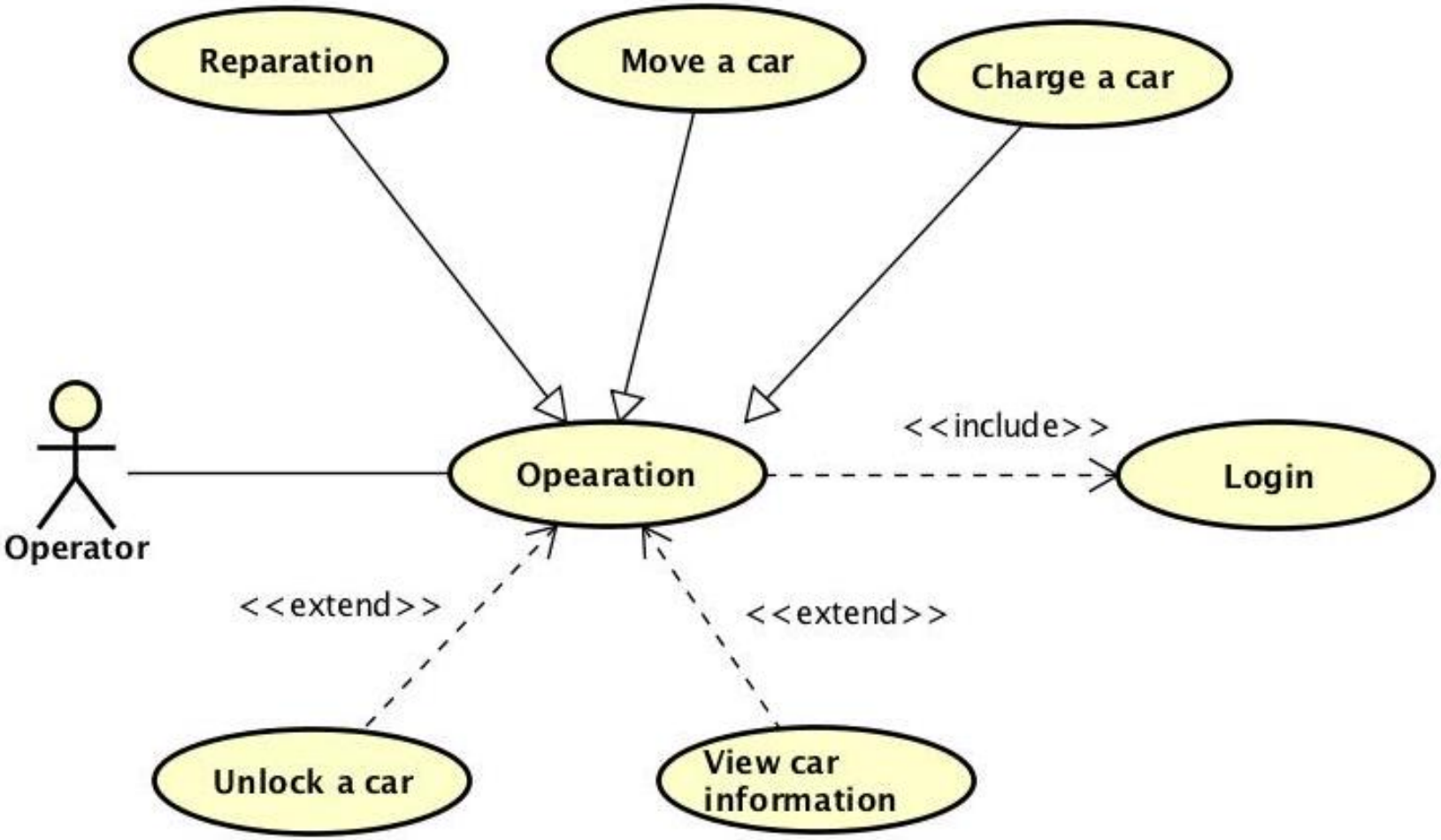
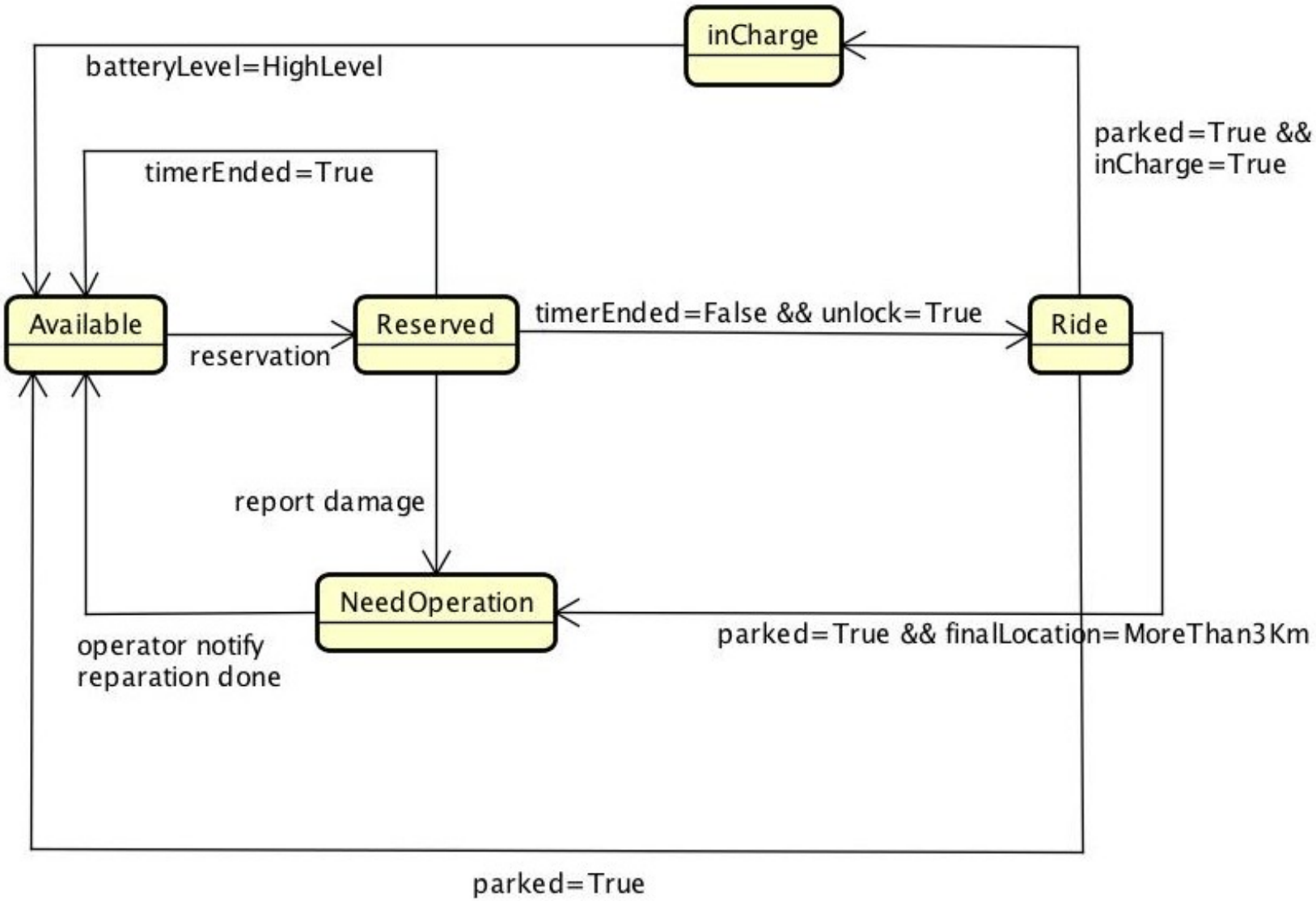

Reparation
Move a car
Charge a car
Operator
Opearation
<<include>>
Login
<<extend>>
<<extend>>
Unlock a car
View car information

Diagramma a stati



Design

Architettura

- Architettura client-server
- Three-tiers application
  - Thin-client
  - Application Logic
  - Database

Architettura

- Architettura client-server
- Three-tiers application
  - Thin-client
  - Application Logic
  - Database

Architettura

- Architettura client-server
- Three-tiers application
  - Thin-client
  - Application Logic
  - Database

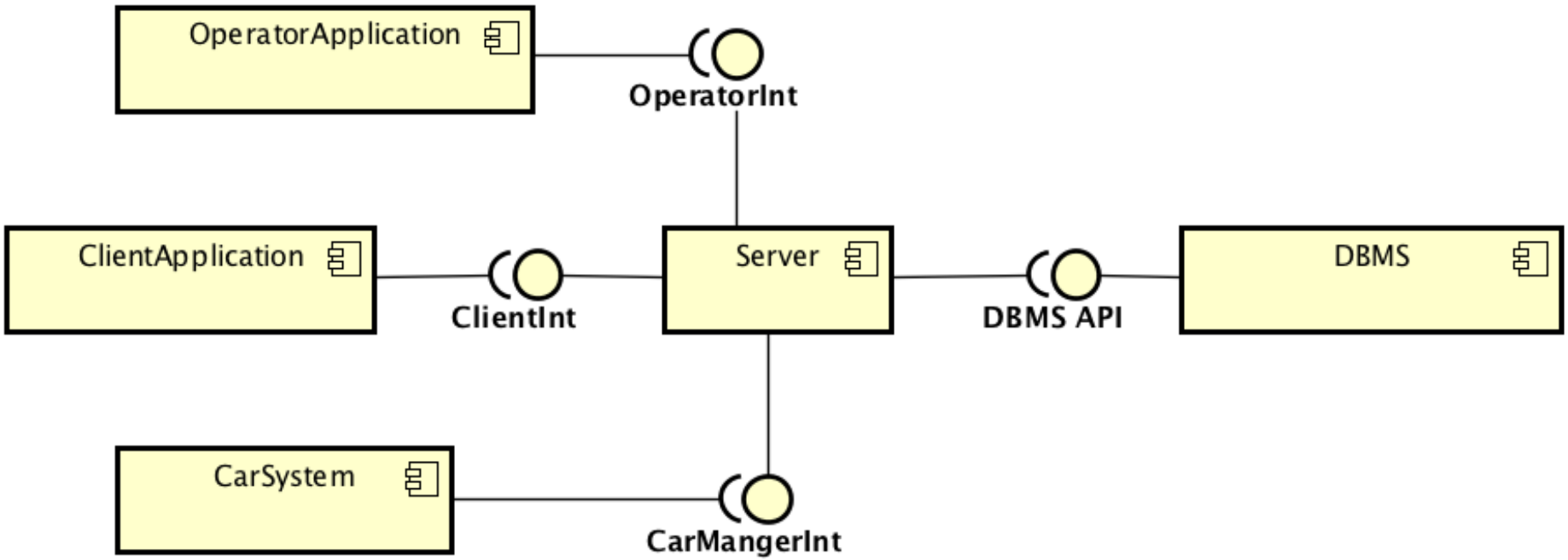
Architettura

- Architettura client-server
- Three-tiers application
  - Thin-client
  - Application Logic
  - Database

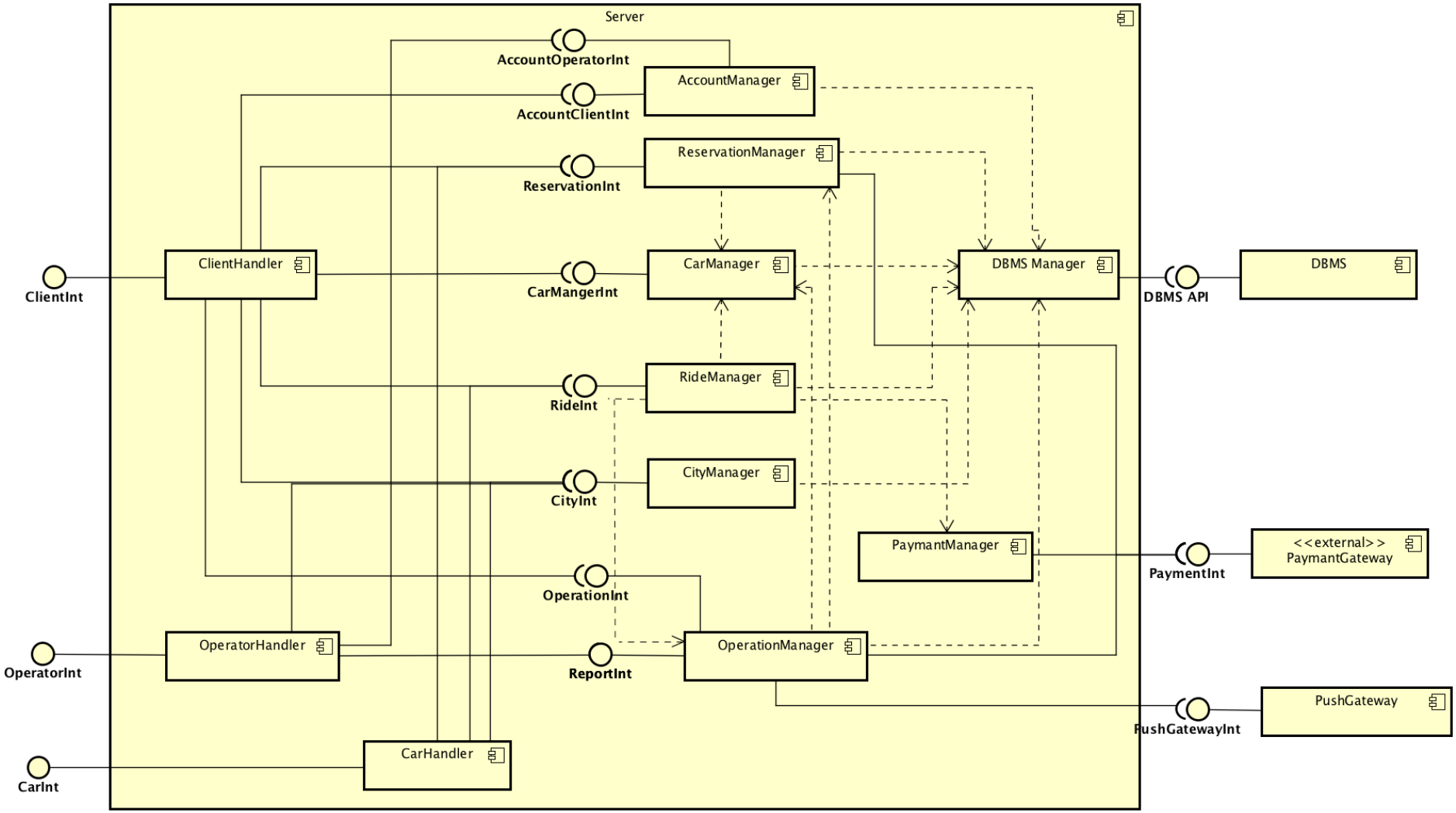
Architettura

- Architettura client-server
- Three-tiers application
  - Thin-client
  - Application Logic
  - Database

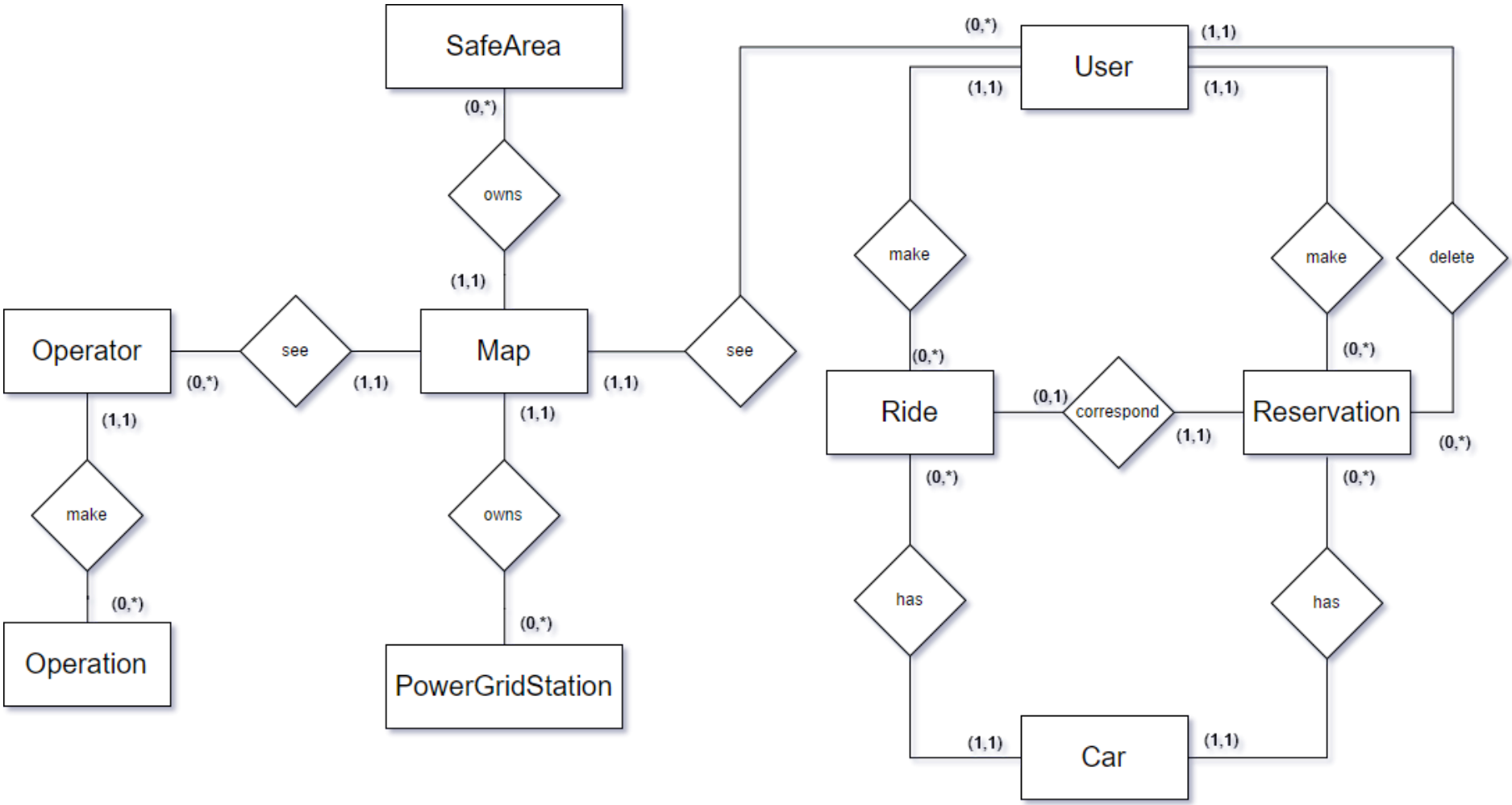
High level components



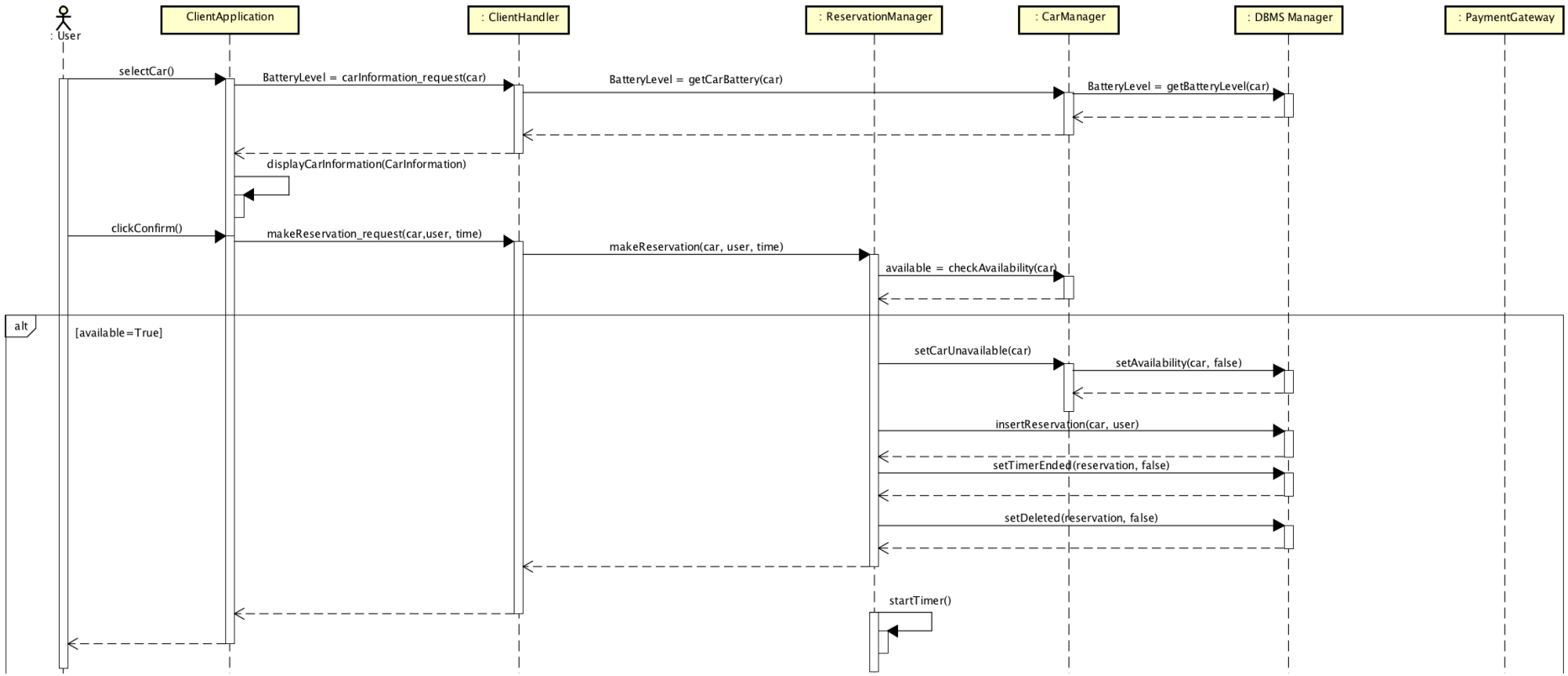
Component view



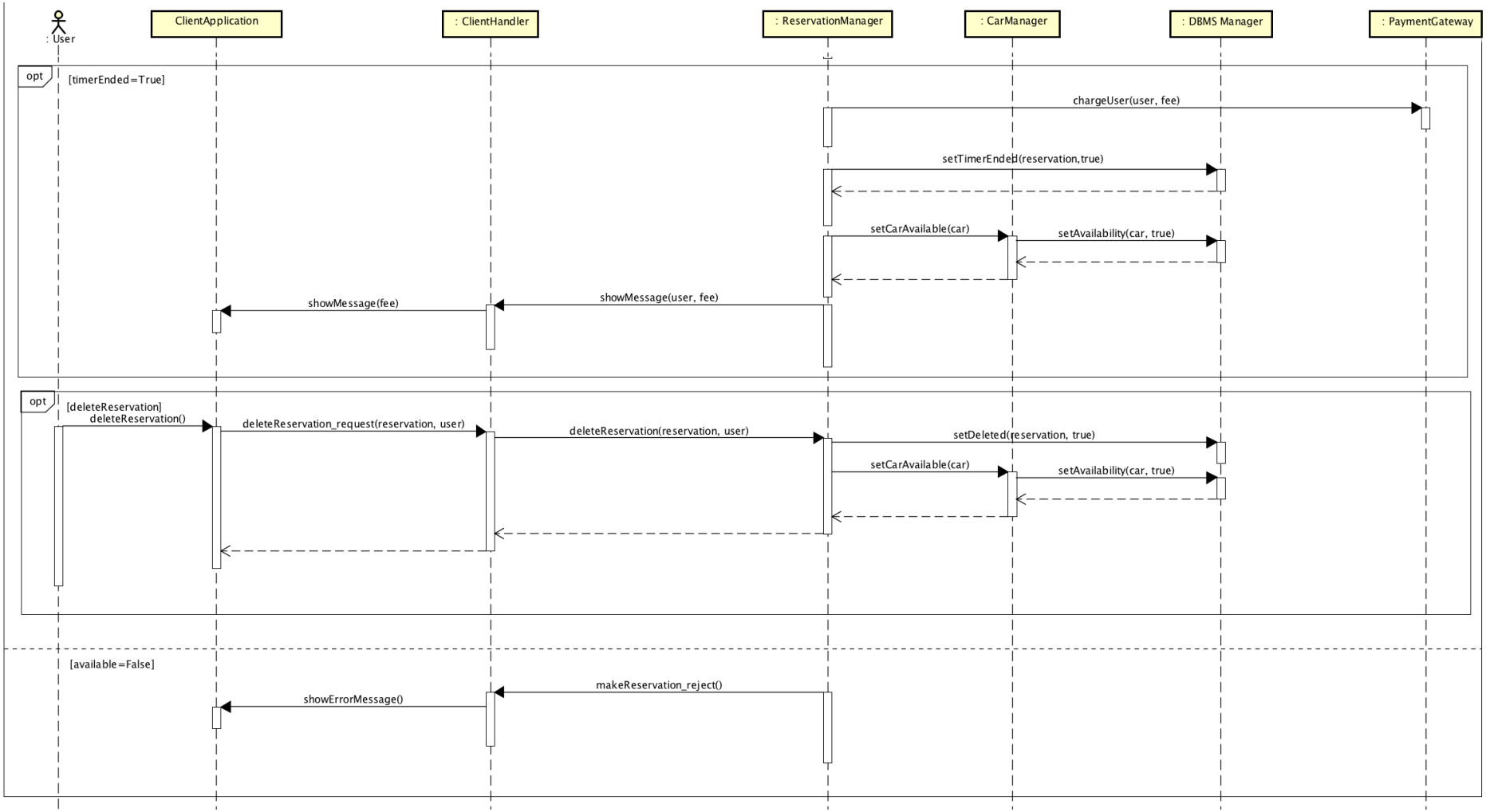
Struttura database



Sequence diagram - Prenotazione 1/2

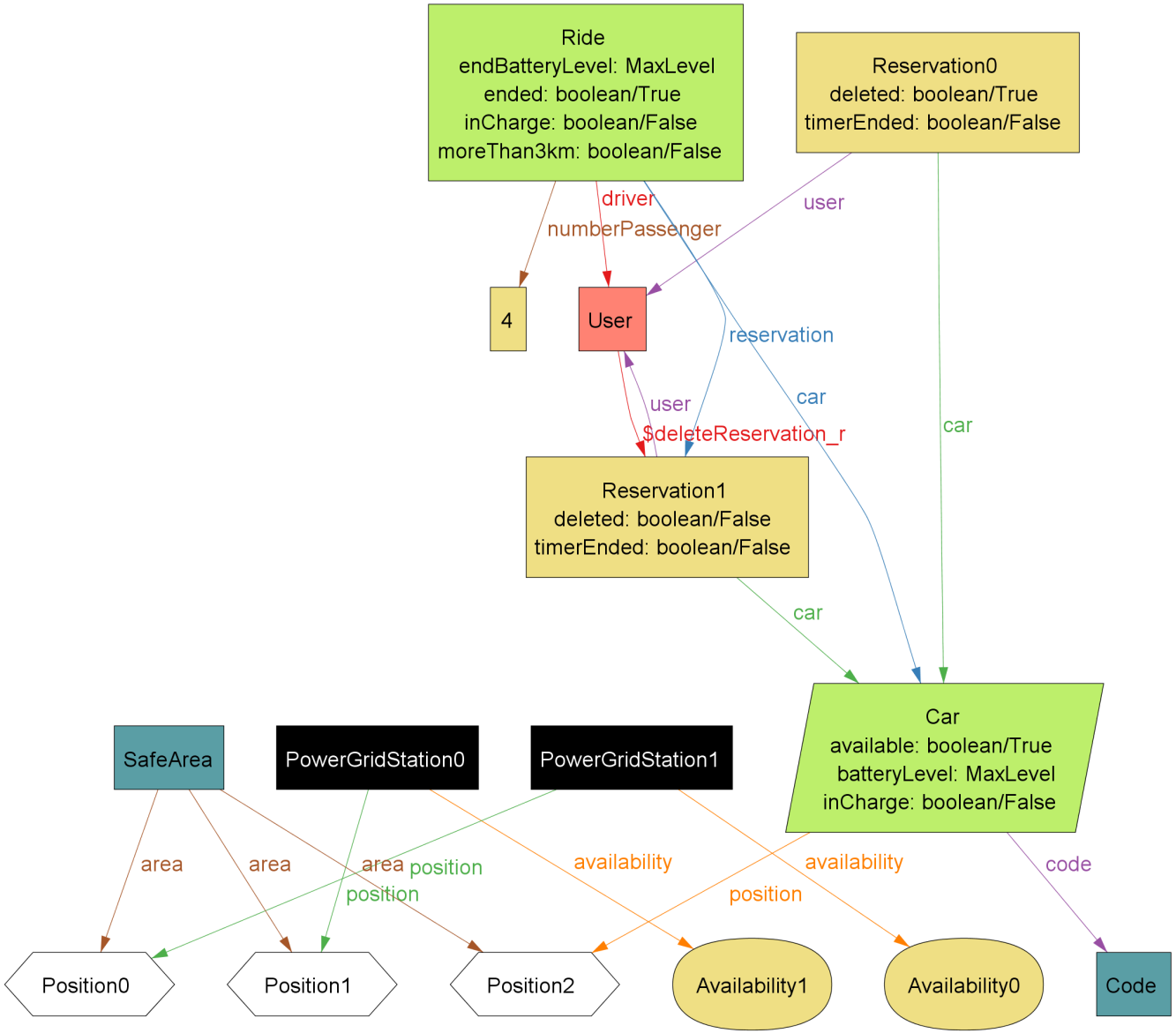


Sequence diagram - Prenotazione 2/2



Alloy - Cancellazione prenotazione

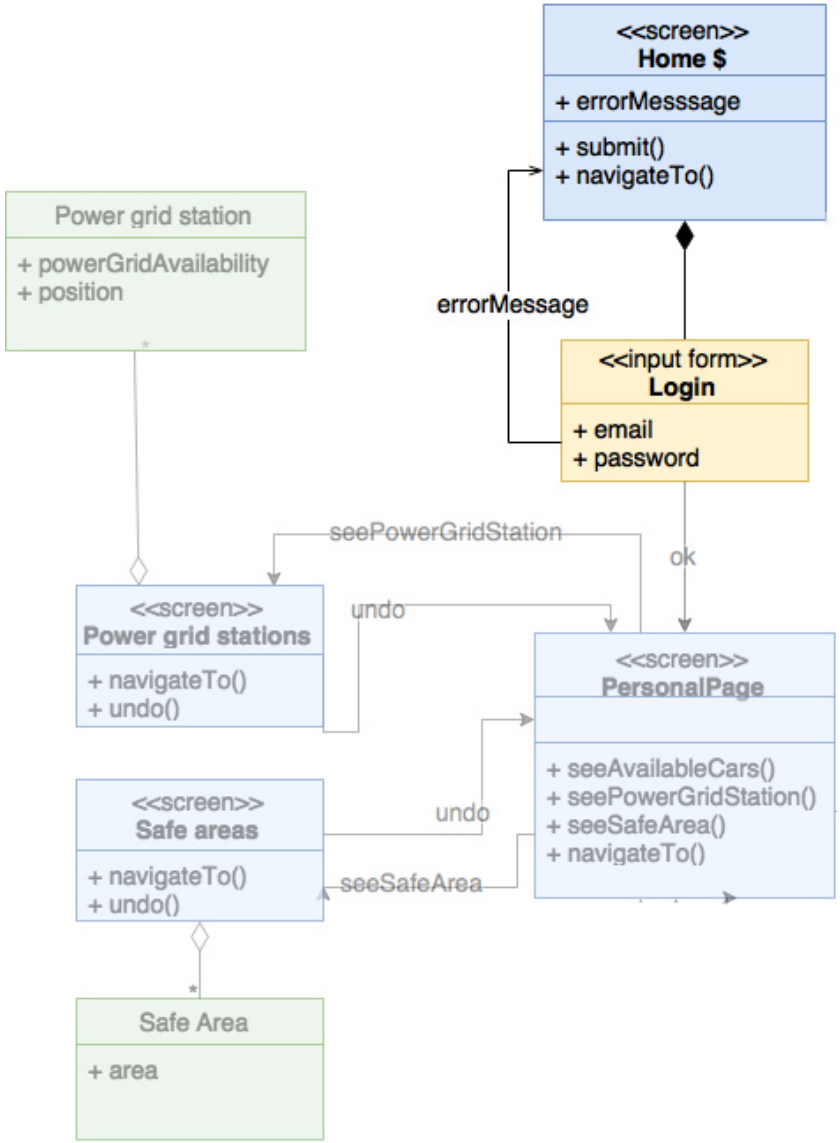
$deleteReservation_r: 1
area: 3
availability: 2
car: 2
car: 1
code: 1
driver: 1
numberPassenger: 1
position: 1
position: 2
reservation: 1
user: 2

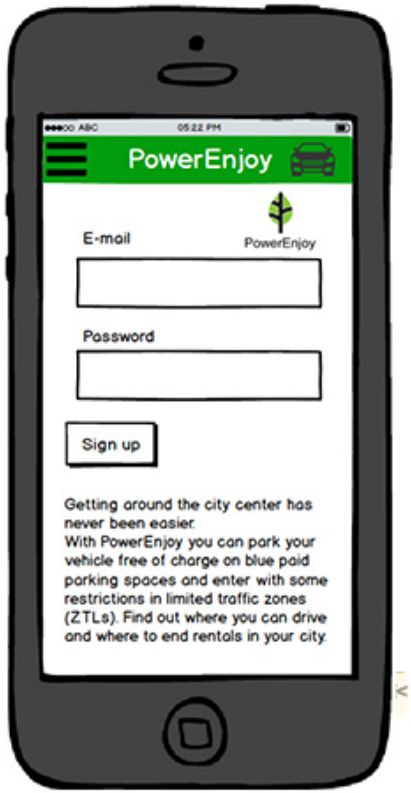


Scenario

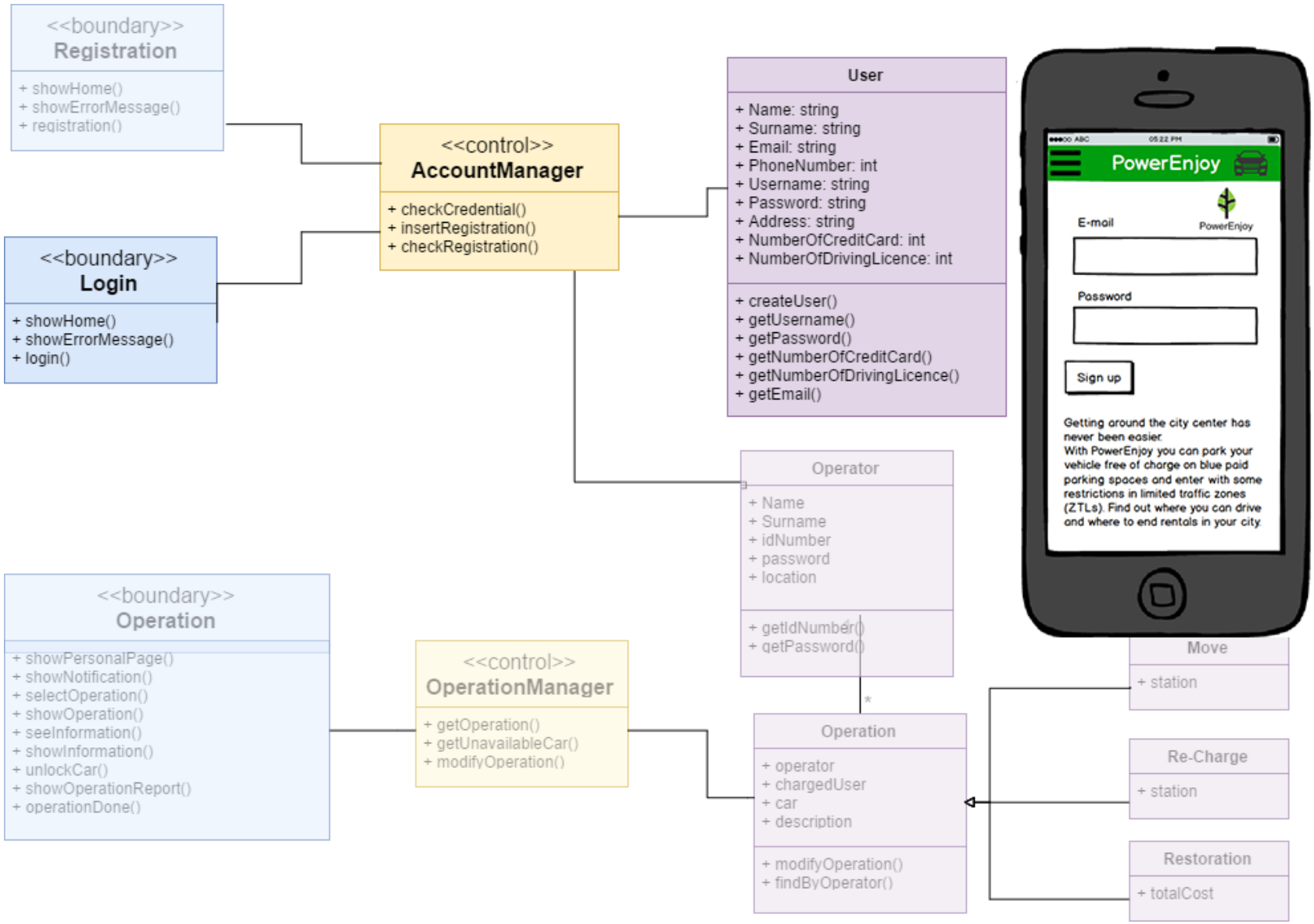
"Francesco è uscito alla sera con gli amici e ha fatto tardi, quindi accede all'applicazione di PowerEnJoy e attiva il gps per vedere se ci sono macchine nelle vicinanze. Francesco ne trova una poco distante dalla sua posizione e la prenota. Una volta avvicinato la sblocca inserendo il codice presente sul cruscotto dell'automobile e inizia il suo noleggio."

UX Login

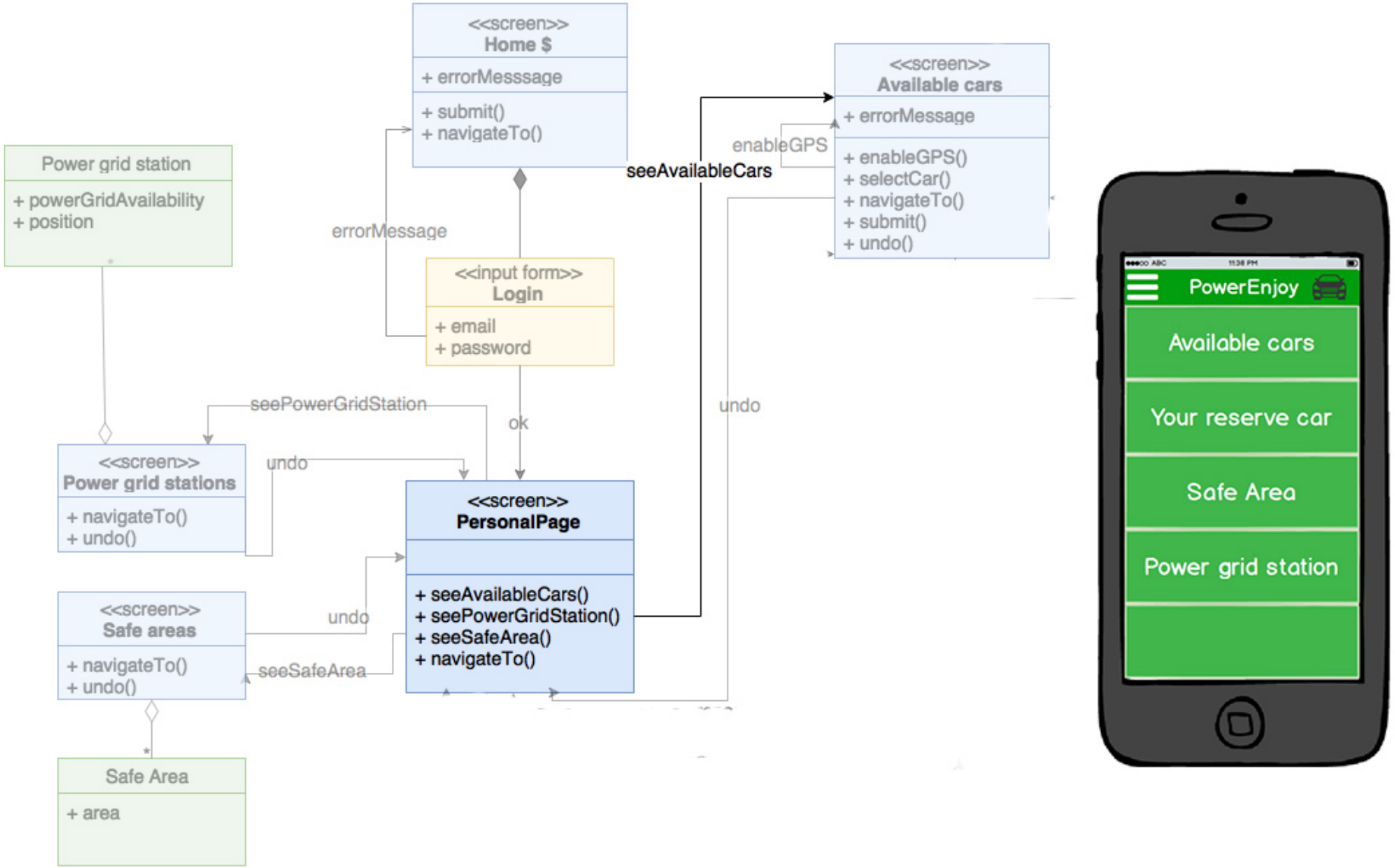

<<screen>>
Home $
+ errorMesssage
+ submit()
+ navigateTo()
errorMessage
<<input form>>
Login
+ email
+ password
Power grid station
+ powerGridAvailability
+ position
seePowerGridStation
ok
undo
<<screen>>
Power grid stations
+ navigateTo()
+ undo()
<<screen>>
PersonalPage
+ seeAvailableCars()
+ seePowerGridStation()
+ seeSafeArea()
+ navigateTo()
undo
<<screen>>
Safe areas
+ navigateTo()
+ undo()
seeSafeArea
Safe Area
+ area


PowerEnjoy
E-mail
PowerEnjoy
Password
Sign up
Getting around the city center has never been easier.
With PowerEnjoy you can park your vehicle free of charge on blue paid parking spaces and enter with some restrictions in limited traffic zones (ZTLs). Find out where you can drive and where to end rentals in your city.

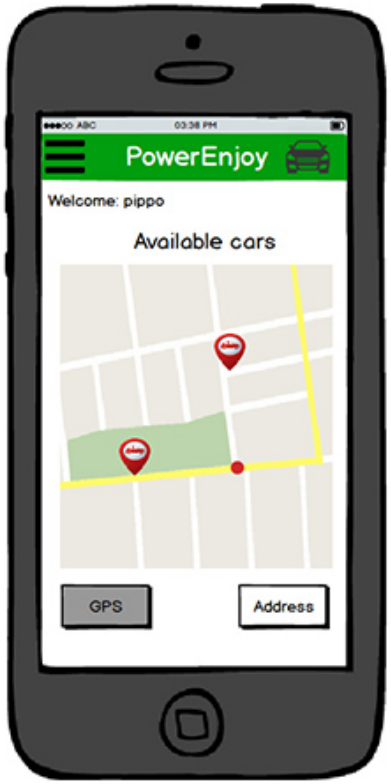
BCE Login

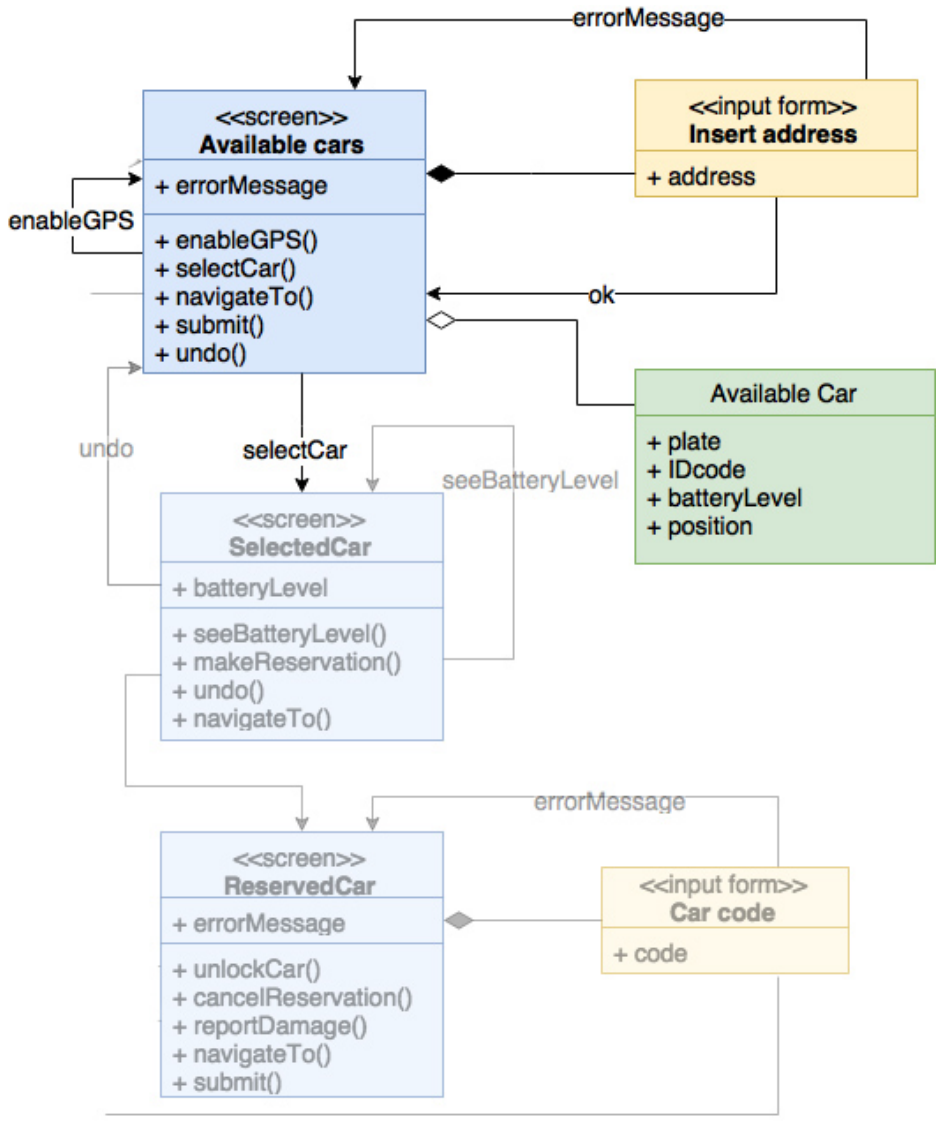

<<boundary>>
Registration
+ showHome()
+ showErrorMessage()
+ registration()
<<boundary>>
Login
+ showHome()
+ showErrorMessage()
+ login()
<<control>>
AccountManager
+ checkCredential()
+ insertRegistration()
+ checkRegistration()
User
+ Name: string
+ Surname: string
+ Email: string
+ PhoneNumber: int
+ Username: string
+ Password: string
+ Address: string
+ NumberOfCreditCard: int
+ NumberOfDrivingLicence: int
+ createUser()
+ getUsername()
+ getPassword()
+ getNumberOfCreditCard()
+ getNumberOfDrivingLicence()
+ getEmail()
PowerEnjoy
E-mail
Password
Sign up
Getting around the city center has never been easier.
With PowerEnjoy you can park your vehicle free of charge on blue paid parking spaces and enter with some restrictions in limited traffic zones (ZTLs). Find out where you can drive and where to end rentals in your city.
Operator
+ Name
+ Surname
+ idNumber
+ password
+ location
+ getIdNumber()
+ getPassword()
<<boundary>>
Operation
+ showPersonalPage()
+ showNotification()
+ selectOperation()
+ showOperation()
+ seeInformation()
+ showInformation()
+ unlockCar()
+ showOperationReport()
+ operationDone()
<<control>>
OperationManager
+ getOperation()
+ getUnavailableCar()
+ modifyOperation()
Operation
+ operator
+ chargedUser
+ car
+ description
+ modifyOperation()
+ findByOperator()
Move
+ station
Re-Charge
+ station
Restoration
+ totalCost

UX Pagina personale

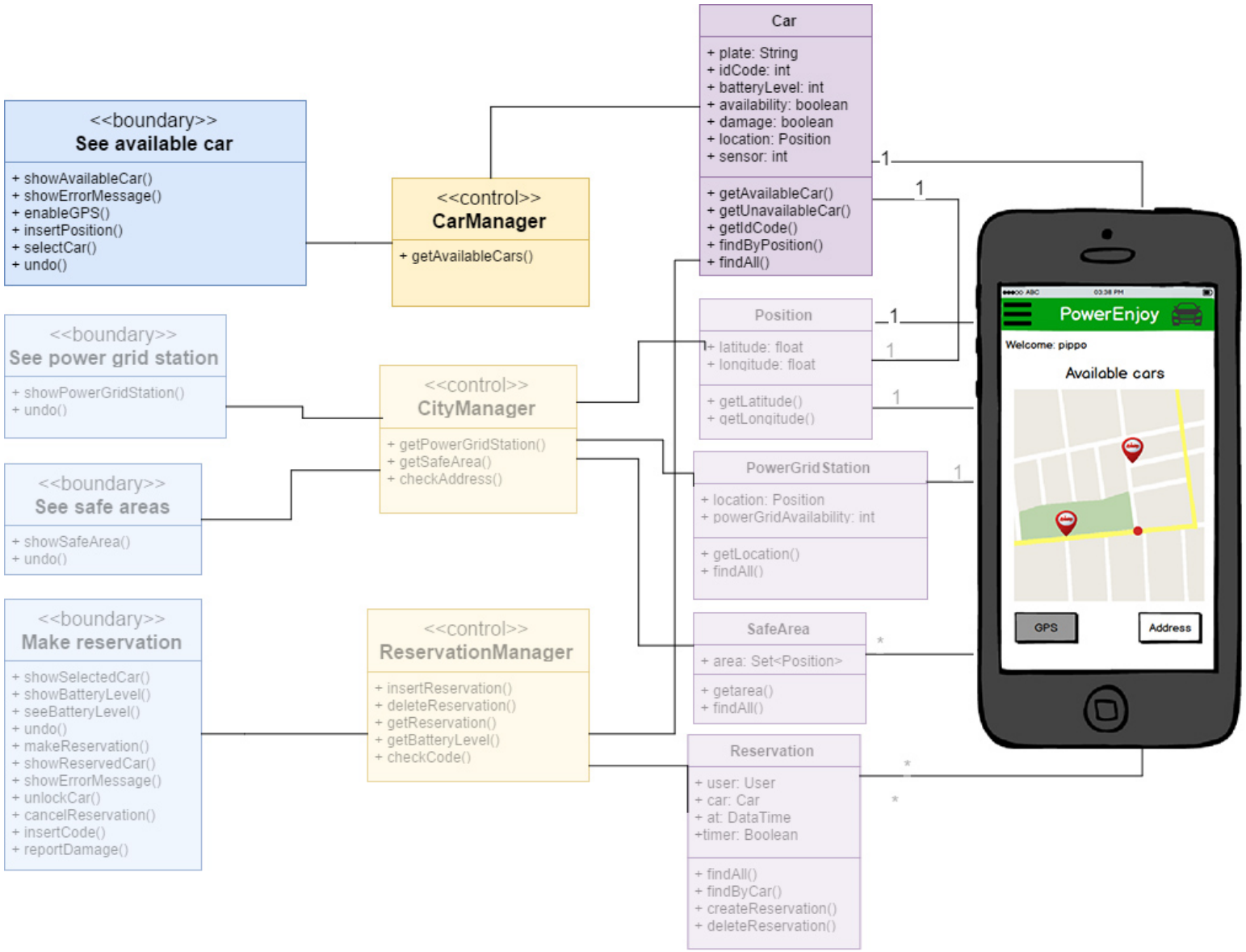


UX Ricerca macchine disponibili

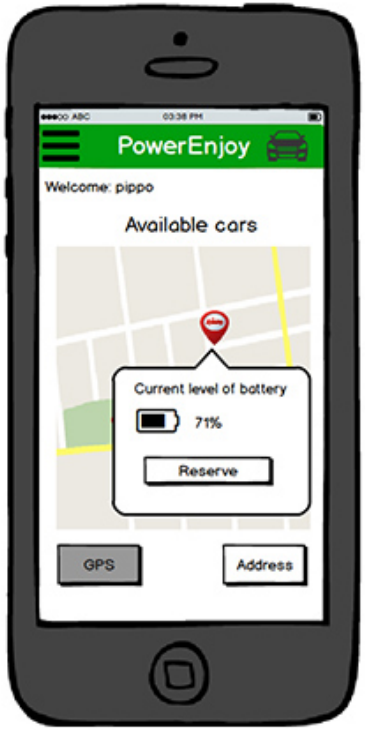


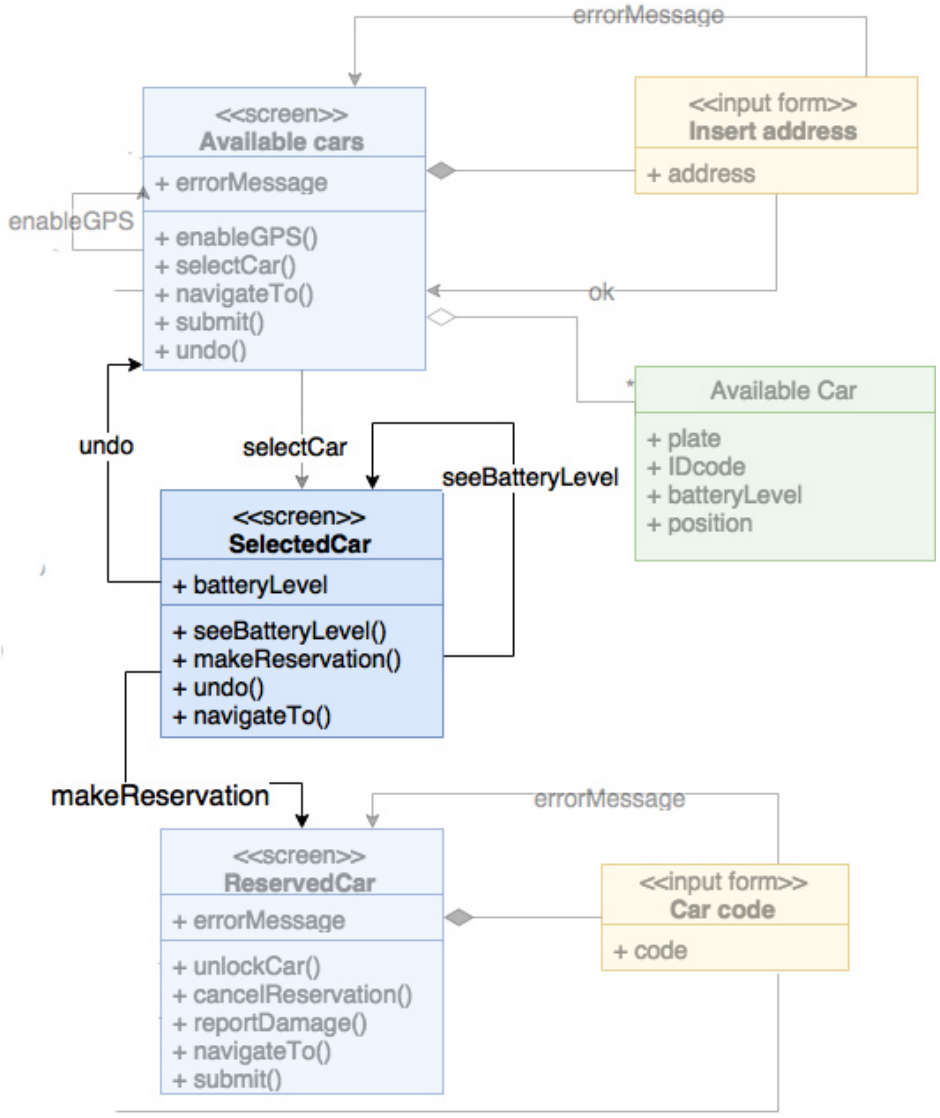


BCE Ricerca macchine disponibili

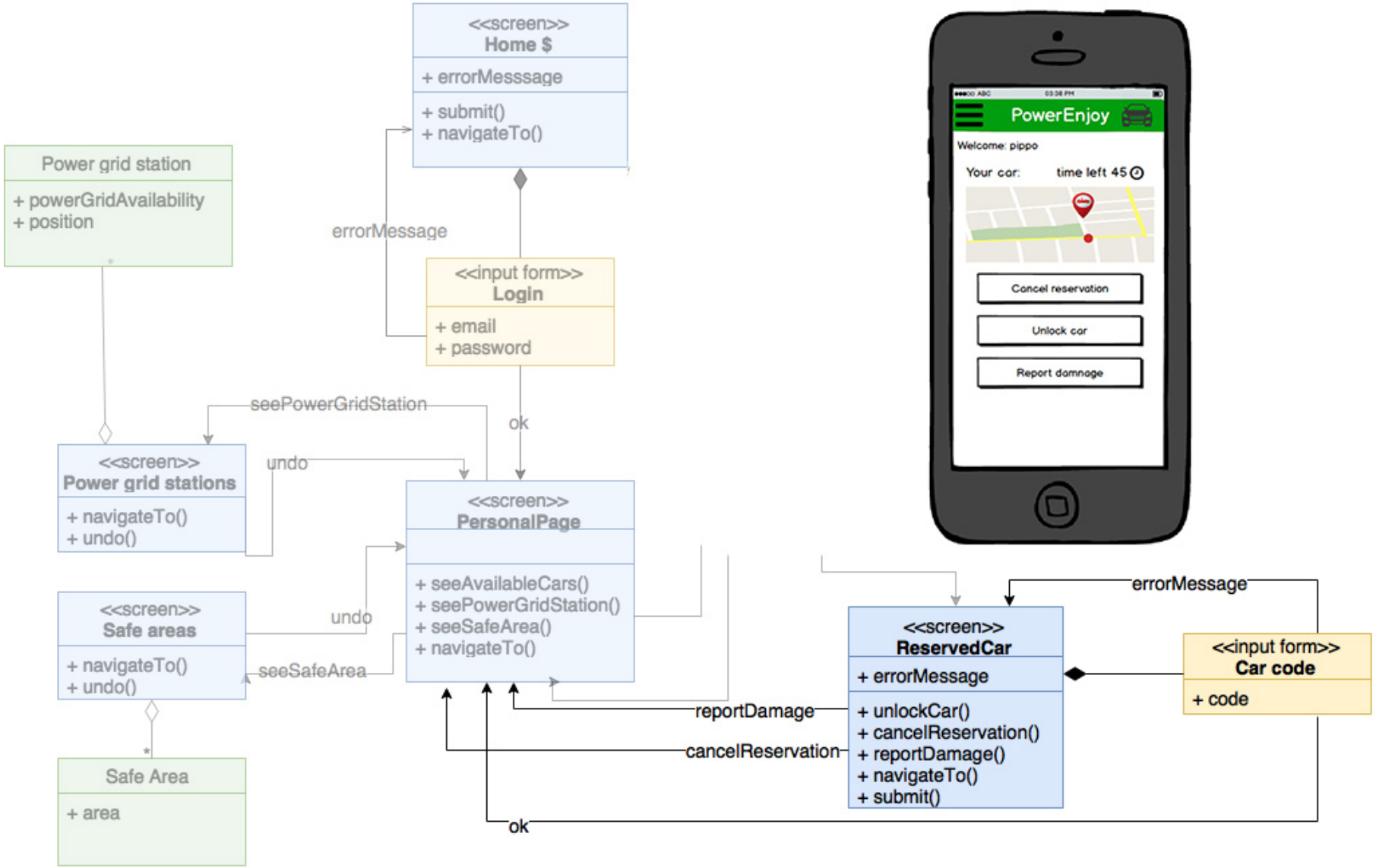


UX Macchina selezionata

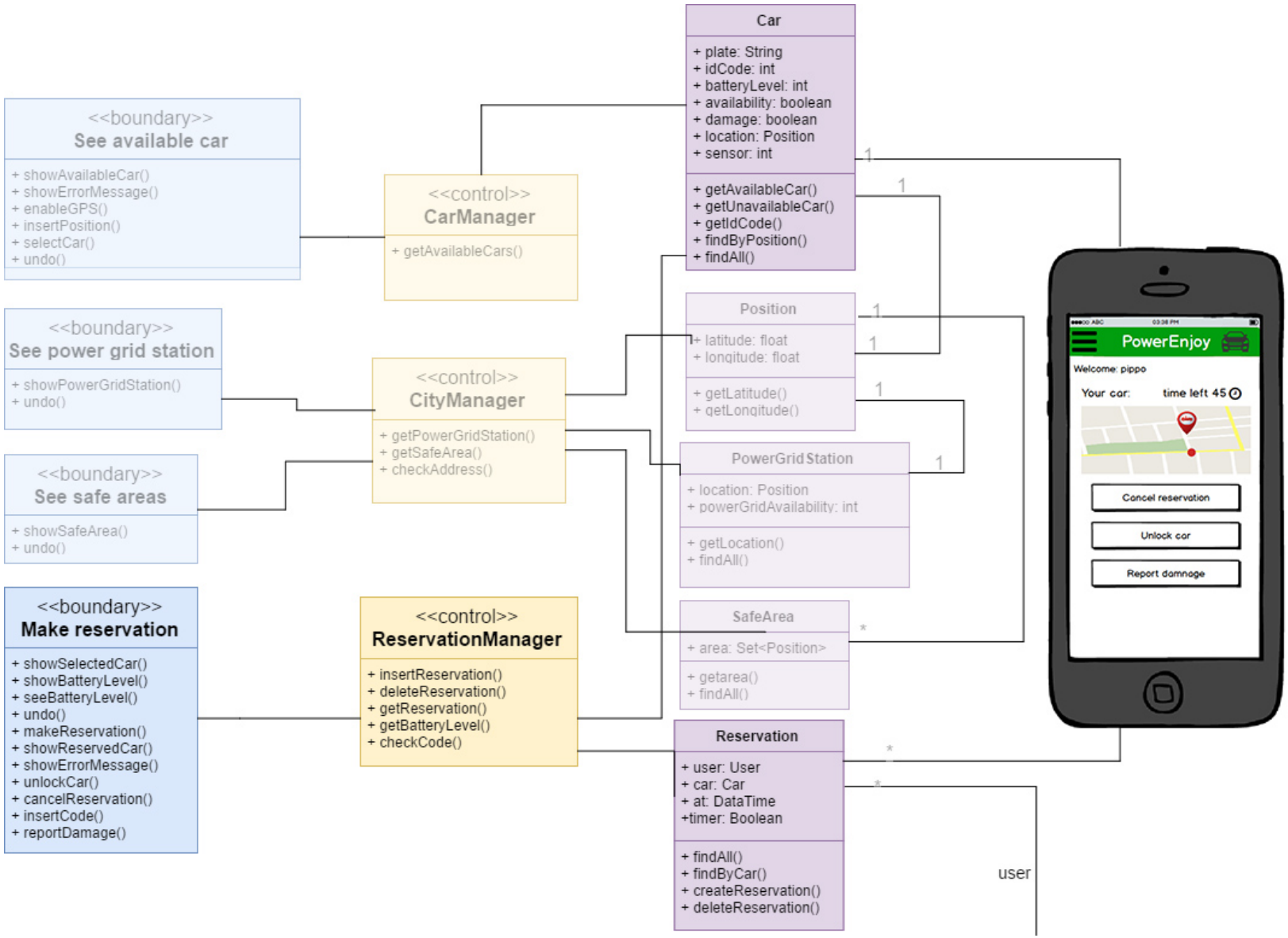




UX Prenotazione



BCE Prenotazione



Test Plan

Prerequisiti

Affinchè sia possibile delineare una strategia di integrazione è necessario che:

- **RASD** e **DD** siano stati precedentemente redatti.
- tutte le classi di ogni component siano correttamente **documentate**.
- ogni classe sia stata testata attraverso i **test d'unità**.

Prerequisiti

Affinchè sia possibile delineare una strategia di integrazione è necessario che:

- **RASD** e **DD** siano stati precedentemente redatti.
- tutte le classi di ogni component siano correttamente **documentate**.
- ogni classe sia stata testata attraverso i **test d'unità**.

Prerequisiti

Affinchè sia possibile delineare una strategia di integrazione è necessario che:

- **RASD** e **DD** siano stati precedentemente redatti.
- tutte le classi di ogni component siano correttamente **documentate**.
- ogni classe sia stata testata attraverso i **test d'unità**.

Strategia di Integrazione

Nei test di integrazione abbiamo utilizzato l'approcio **Bottom-up**, nel quale

- i componenti indipendenti devono essere già stati sviluppati
- l'unico stub necessario è quello che simula il comportamento del Payment Gateway
- i driver sono temporaneamente indispendabili

Strategia di Integrazione

Nei test di integrazione abbiamo utilizzato l'approcio **Bottom-up**, nel quale

- i componenti indipendenti devono essere già stati sviluppati
- l'unico stub necessario è quello che simula il comportamento del Payment Gateway
- i driver sono temporaneamente indispendabili

Strategia di Integrazione

Nei test di integrazione abbiamo utilizzato l'approcio **Bottom-up**, nel quale

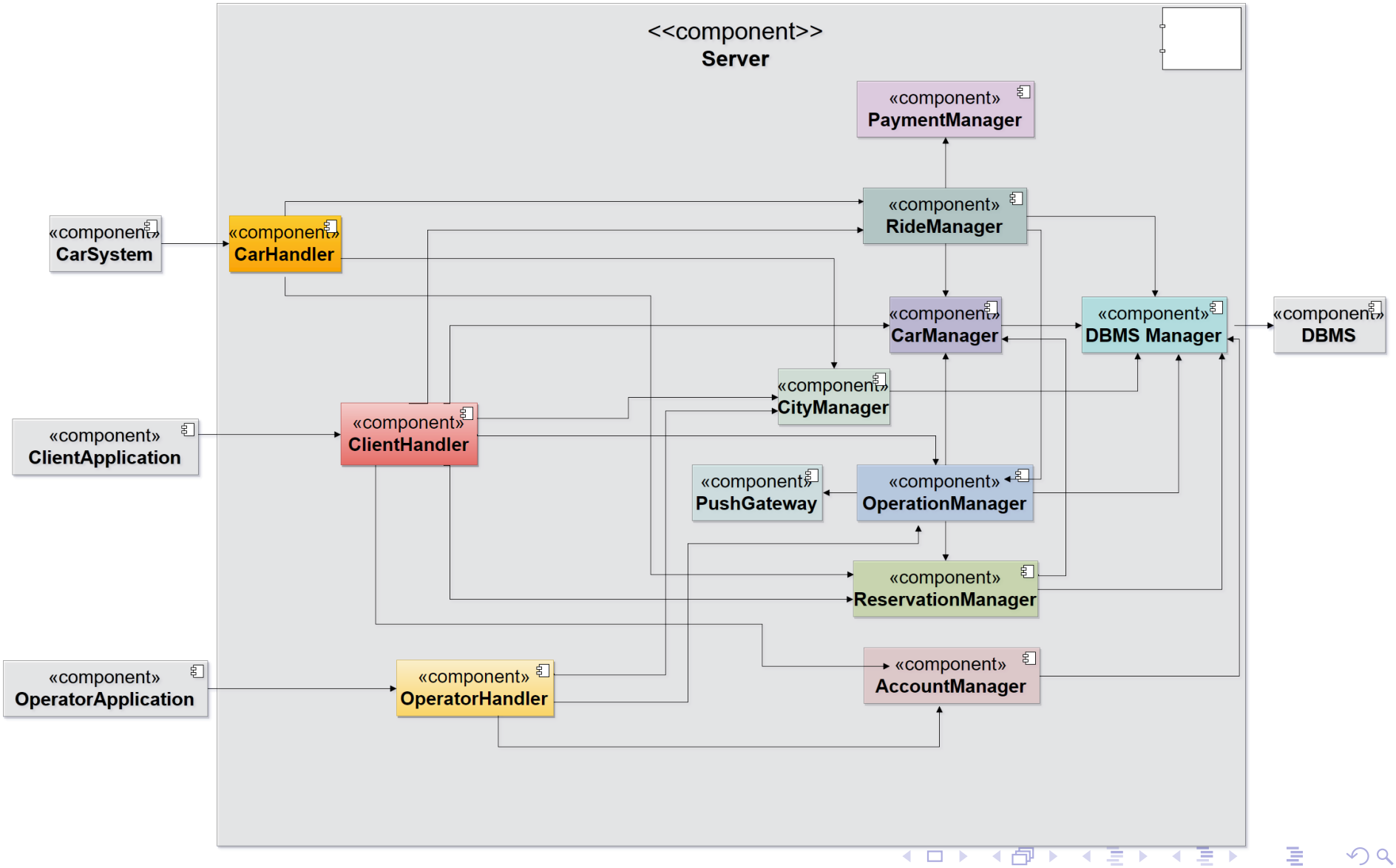
- i componenti indipendenti devono essere già stati sviluppati
- l'unico stub necessario è quello che simula il comportamento del Payment Gateway
- i driver sono temporaneamente indispendabili

Strategia di Integrazione

Nei test di integrazione abbiamo utilizzato l'approcio **Bottom-up**, nel quale

- i componenti indipendenti devono essere già stati sviluppati
- l'unico stub necessario è quello che simula il comportamento del Payment Gateway
- i driver sono temporaneamente indispendabili

Integrazione dei componenti



Tools

- **JUnit**
  - framework open source con lo scopo di scrivere ed eseguire test di unità
  - permette agli sviluppatori di creare incrementalmente test suits
  - può essere utilizzato insieme a **Mockito**
- Mockito
  - permette di creare e configurare mock objects
  - semplifica il test di classi con dipendenze esterne
  - utile per la creazione di stubs
- JMeter
  - open source software utilizzato per il test delle performance
  - simula un gruppo di utenti che inviano richieste al server

Tools

- **JUnit**
  - framework open source con lo scopo di scrivere ed eseguire test di unità
  - permette agli sviluppatori di creare incrementalmente test suits
  - può essere utilizzato insieme a **Mockito**
- Mockito
  - permette di creare e configurare mock objects
  - semplifica il test di classi con dipendenze esterne
  - utile per la creazione di stubs
- JMeter
  - open source software utilizzato per il test delle performance
  - simula un gruppo di utenti che inviano richieste al server

Tools

- **JUnit**
  - framework open source con lo scopo di scrivere ed eseguire test di unità
  - permette agli sviluppatori di creare incrementalmente test suits
  - può essere utilizzato insieme a **Mockito**
- **Mockito**
  - permette di creare e configurare mock objects
  - semplifica il test di classi con dipendenze esterne
  - utile per la creazione di stubs
- **JMeter**
  - open source software utilizzato per il test delle performance
  - simula un gruppo di utenti che inviano richieste al server

Tools

- **JUnit**
  - framework open source con lo scopo di scrivere ed eseguire test di unità
  - permette agli sviluppatori di creare incrementalmente test suits
  - può essere utilizzato insieme a **Mockito**
- Mockito
  - permette di creare e configurare mock objects
  - semplifica il test di classi con dipendenze esterne
  - utile per la creazione di stubs
- JMeter
  - open source software utilizzato per il test delle performance
  - simula un gruppo di utenti che inviano richieste al server

Tools

- **JUnit**
  - framework open source con lo scopo di scrivere ed eseguire test di unità
  - permette agli sviluppatori di creare incrementalmente test suits
  - può essere utilizzato insieme a **Mockito**
- **Mockito**
  - permette di creare e configurare mock objects
  - semplifica il test di classi con dipendenze esterne
  - utile per la creazione di stubs
- JMeter
  - open source software utilizzato per il test delle performance
  - simula un gruppo di utenti che inviano richieste al server

Tools

- **JUnit**
  - framework open source con lo scopo di scrivere ed eseguire test di unità
  - permette agli sviluppatori di creare incrementalmente test suits
  - può essere utilizzato insieme a **Mockito**
- **Mockito**
  - permette di creare e configurare mock objects
  - semplifica il test di classi con dipendenze esterne
  - utile per la creazione di stubs
- JMeter
  - open source software utilizzato per il test delle performance
  - simula un gruppo di utenti che inviano richieste al server

Tools

- **JUnit**
  - framework open source con lo scopo di scrivere ed eseguire test di unità
  - permette agli sviluppatori di creare incrementalmente test suits
  - può essere utilizzato insieme a **Mockito**
- **Mockito**
  - permette di creare e configurare mock objects
  - semplifica il test di classi con dipendenze esterne
  - utile per la creazione di stubs
- JMeter
  - open source software utilizzato per il test delle performance
  - simula un gruppo di utenti che inviano richieste al server

Tools

- **JUnit**
  - framework open source con lo scopo di scrivere ed eseguire test di unità
  - permette agli sviluppatori di creare incrementalmente test suits
  - può essere utilizzato insieme a **Mockito**
- **Mockito**
  - permette di creare e configurare mock objects
  - semplifica il test di classi con dipendenze esterne
  - utile per la creazione di stubs
- JMeter
  - open source software utilizzato per il test delle performance
  - simula un gruppo di utenti che inviano richieste al server

Tools

- **JUnit**
  - framework open source con lo scopo di scrivere ed eseguire test di unità
  - permette agli sviluppatori di creare incrementalmente test suits
  - può essere utilizzato insieme a **Mockito**
- **Mockito**
  - permette di creare e configurare mock objects
  - semplifica il test di classi con dipendenze esterne
  - utile per la creazione di stubs
- **JMeter**
  - open source software utilizzato per il test delle performance
  - simula un gruppo di utenti che inviano richieste al server

Tools

- **JUnit**
  - framework open source con lo scopo di scrivere ed eseguire test di unità
  - permette agli sviluppatori di creare incrementalmente test suits
  - può essere utilizzato insieme a **Mockito**
- **Mockito**
  - permette di creare e configurare mock objects
  - semplifica il test di classi con dipendenze esterne
  - utile per la creazione di stubs
- **JMeter**
  - open source software utilizzato per il test delle performance
  - simula un gruppo di utenti che inviano richieste al server

Tools

- **JUnit**
  - framework open source con lo scopo di scrivere ed eseguire test di unità
  - permette agli sviluppatori di creare incrementalmente test suits
  - può essere utilizzato insieme a **Mockito**
- **Mockito**
  - permette di creare e configurare mock objects
  - semplifica il test di classi con dipendenze esterne
  - utile per la creazione di stubs
- **JMeter**
  - open source software utilizzato per il test delle performance
  - simula un gruppo di utenti che inviano richieste al server

Project plan

Function point

- Un **function point** è un'unità di misura per esprimere le dimensioni del software
- La dimensione del software è funzione delle caratteristiche del programma
  - External Input
  - External Output
  - External Inquiry
  - Internal Logic File
  - External Logic Files
- $UFP = \sum(N_i * w_i)$, dove N_i è il numero di elementi di un determinato tipo e w_i il peso associato.

Function point

- Un **function point** è un'unità di misura per esprimere le dimensioni del software
- La dimensione del software è funzione delle caratteristiche del programma
  - External Input
  - External Output
  - External Inquiry
  - Internal Logic File
  - External Logic Files
- $UFP = \sum(N_i * w_i)$, dove N_i è il numero di elementi di un determinato tipo e w_i il peso associato.

Function point

- Un **function point** è un'unità di misura per esprimere le dimensioni del software
- La dimensione del software è funzione delle caratteristiche del programma
  - External Input
  - External Output
  - External Inquiry
  - Internal Logic File
  - External Logic Files
- $UFP = \sum(N_i * w_i)$, dove N_i è il numero di elementi di un determinato tipo e w_i il peso associato.

Function point

- Un **function point** è un'unità di misura per esprimere le dimensioni del software
- La dimensione del software è funzione delle caratteristiche del programma
  - External Input
  - External Output
  - External Inquiry
  - Internal Logic File
  - External Logic Files
- $UFP = \sum(N_i * w_i)$, dove N_i è il numero di elementi di un determinato tipo e w_i il peso associato.

Function point

- Un **function point** è un'unità di misura per esprimere le dimensioni del software
- La dimensione del software è funzione delle caratteristiche del programma
  - External Input
  - External Output
  - External Inquiry
  - Internal Logic File
  - External Logic Files
- $UFP = \sum(N_i * w_i)$, dove N_i è il numero di elementi di un determinato tipo e w_i il peso associato.

Function point

- Un **function point** è un'unità di misura per esprimere le dimensioni del software
- La dimensione del software è funzione delle caratteristiche del programma
  - External Input
  - External Output
  - External Inquiry
  - Internal Logic File
  - External Logic Files
- $UFP = \sum(N_i * w_i)$, dove N_i è il numero di elementi di un determinato tipo e w_i il peso associato.

Function point

- Un **function point** è un'unità di misura per esprimere le dimensioni del software
- La dimensione del software è funzione delle caratteristiche del programma
  - External Input
  - External Output
  - External Inquiry
  - Internal Logic File
  - External Logic Files
- $UFP = \sum(N_i * w_i)$, dove N_i è il numero di elementi di un determinato tipo e w_i il peso associato.

Function point

- Un **function point** è un'unità di misura per esprimere le dimensioni del software
- La dimensione del software è funzione delle caratteristiche del programma
  - External Input
  - External Output
  - External Inquiry
  - Internal Logic File
  - External Logic Files
- $UFP = \sum(N_i * w_i)$, dove N_i è il numero di elementi di un determinato tipo e w_i il peso associato.

Internal Logic File

Internal Logic File	**Complexity**	**FPs**
Information registerd user	Simple	7
Information registered operator	Simple	7
Safe area	Medium	10
Power grid stations	Simple	7
Map	Complex	15
Car informations	Simple	7
Reservation	Simple	7
Ride	Simple	7
Operations	Medium	10
Total		77

SLOC

Function Type	**Function Complexity**
External Input (EI)	41
External Output (EO)	26
External Inquiry (EQ)	27
Internal Logical File (ILF)	77
External Interface Files (EIF)	30
Total	201

$$SLOC = 46 * 201 = 9246$$

COCOMO II

- **COCOMO II** è stato utilizzato per stimare l'effort e il tempo necessari allo sviluppo di PowerEnjoy
- Post-architecture model
- Per il calcolo dell'effort, ad ogni Scale Driver e Cost Driver è stato assegnato uno rating level tenendo in considerazione le caratteristiche del software che verrà sviluppato

COCOMO II

- **COCOMO II** è stato utilizzato per stimare l'effort e il tempo necessari allo sviluppo di PowerEnjoy
- **Post-architecture model**
- Per il calcolo dell'effort, ad ogni Scale Driver e Cost Driver è stato assegnato uno rating level tenendo in considerazione le caratteristiche del software che verrà sviluppato

COCOMO II

- **COCOMO II** è stato utilizzato per stimare l'effort e il tempo necessari allo sviluppo di PowerEnjoy
- **Post-architecture model**
- Per il calcolo dell'effort, ad ogni Scale Driver e Cost Driver è stato assegnato uno rating level tenendo in considerazione le caratteristiche del software che verrà sviluppato

Effort

$$E = B + 0.01 * \sum_{j=1}^{5} SF_j = 0.91 + 0.01 * 17.85 = 1.0885$$

$$PM = A * Size^E * \prod_{i=1}^{n} EM_i = 2.94 * 9246^{1.0885} * 0.81 \backsimeq 27PM$$

dove:

- $A = 2.94\ ^{PM}/_{KSLOC}$
- $Size$ è la dimensione stimata con i FP
- EM_i sono i Cost Driver
- $B = 0.91$
- SF_j sono i Scale Factor

Schedule

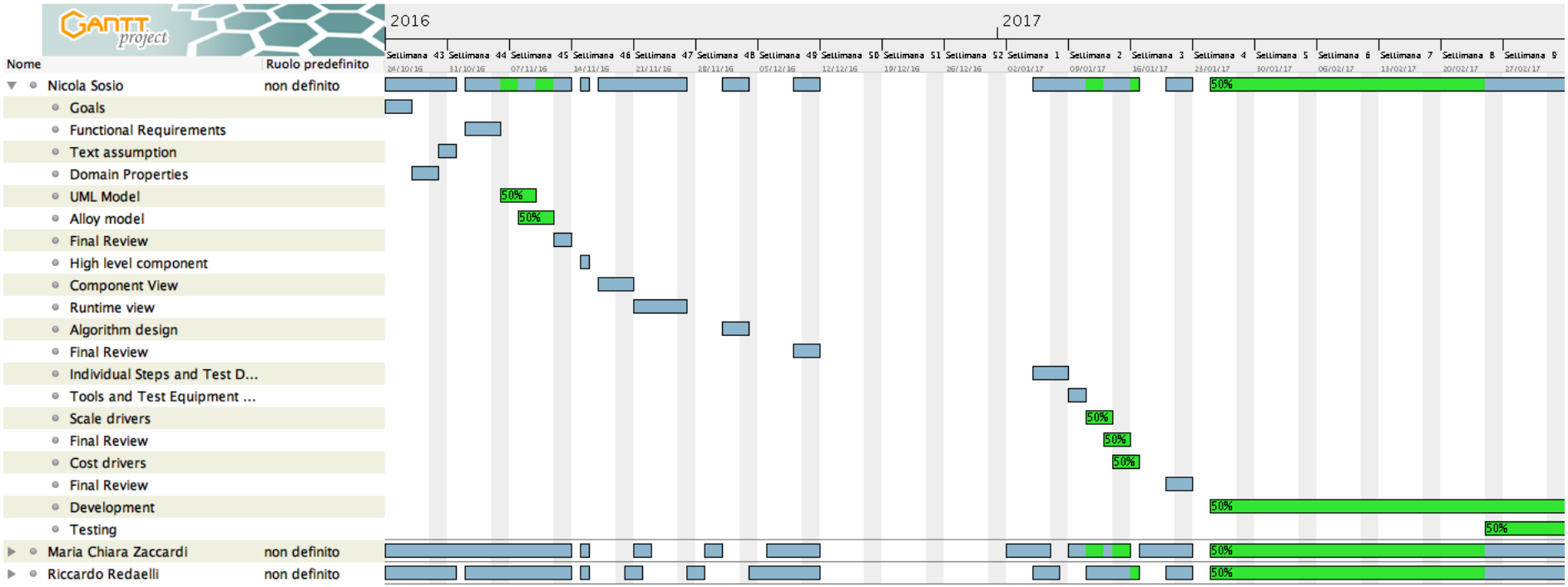
$$TDEV = [C * PM^F] * \frac{SCED\%}{100} = 3.67 * 27^{0.3157} * \frac{130}{100} \simeq 14 months$$

$$F = D + 0.2 * (E - B) = 0.28 + 0.2 * (1.86 - 0.91) = 0.3157$$

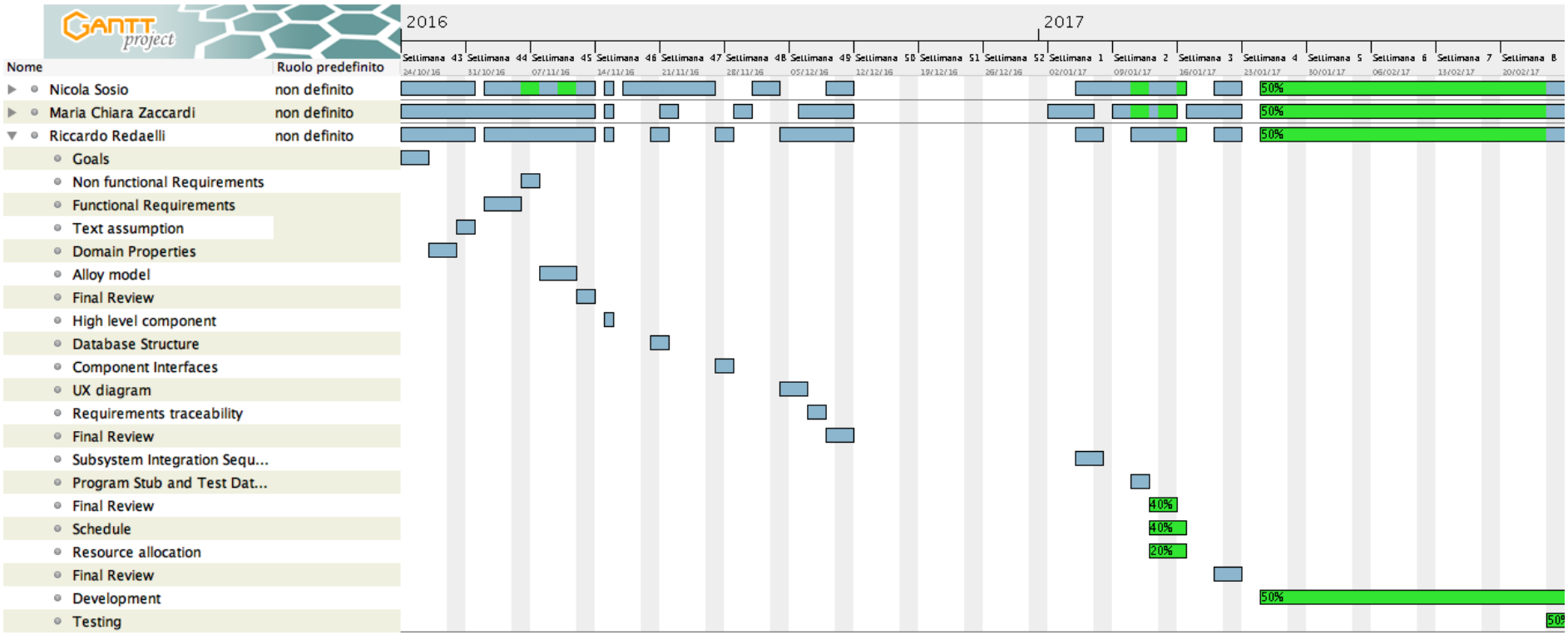
dove:

- $C = 3.67$.
- PM è il numero di persone al mese stimate precedentemente
- $D = 0.28$.
- $B = 0.91$.
- $SCED\%$ è il fattore percentuale di compressione/espansione

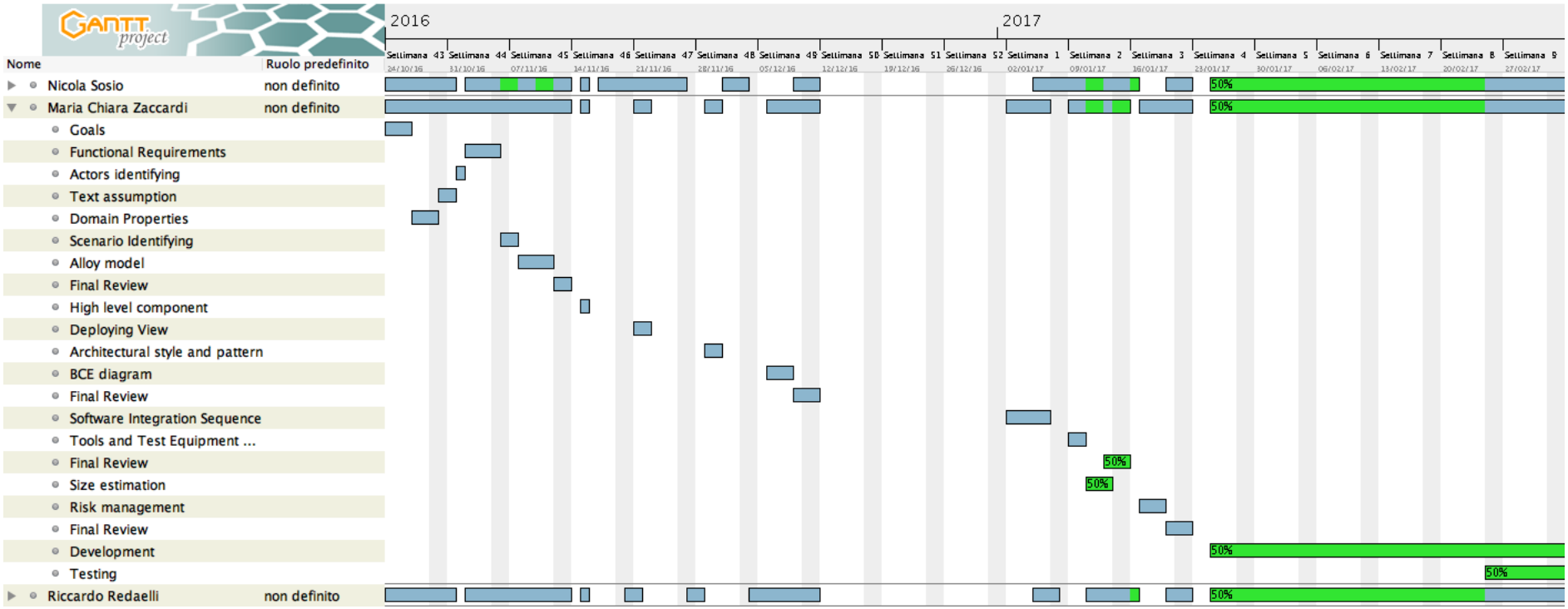
Allocazione risorse 1/3



Allocazione risorse 2/3



Allocazione risorse 3/3



Rischi

I principali rischi che possono incorrere durante lo sviluppo del progetto sono i seguenti:

- cambiamento nei requirements
- perdita di dati
- dipendenza da servizi esterni
- affidabilità e compatibilità dei sensori
- comportamenti inaspettati dei clienti
- concorrenza
- sponsorizzare PowerEnJoy

Rischi

I principali rischi che possono incorrere durante lo sviluppo del progetto sono i seguenti:

- cambiamento nei requirements
- perdita di dati
- dipendenza da servizi esterni
- affidabilità e compatibilità dei sensori
- comportamenti inaspettati dei clienti
- concorrenza
- sponsorizzare PowerEnJoy

Rischi

I principali rischi che possono incorrere durante lo sviluppo del progetto sono i seguenti:

- cambiamento nei requirements
- perdita di dati
- dipendenza da servizi esterni
- affidabilità e compatibilità dei sensori
- comportamenti inaspettati dei clienti
- concorrenza
- sponsorizzare PowerEnJoy

Rischi

I principali rischi che possono incorrere durante lo sviluppo del progetto sono i seguenti:

- cambiamento nei requirements
- perdita di dati
- dipendenza da servizi esterni
- affidabilità e compatibilità dei sensori
- comportamenti inaspettati dei clienti
- concorrenza
- sponsorizzare PowerEnJoy

Rischi

I principali rischi che possono incorrere durante lo sviluppo del progetto sono i seguenti:

- cambiamento nei requirements
- perdita di dati
- dipendenza da servizi esterni
- affidabilità e compatibilità dei sensori
- comportamenti inaspettati dei clienti
- concorrenza
- sponsorizzare PowerEnJoy

Rischi

I principali rischi che possono incorrere durante lo sviluppo del progetto sono i seguenti:

- cambiamento nei requirements
- perdita di dati
- dipendenza da servizi esterni
- affidabilità e compatibilità dei sensori
- comportamenti inaspettati dei clienti
- concorrenza
- sponsorizzare PowerEnJoy

Rischi

I principali rischi che possono incorrere durante lo sviluppo del progetto sono i seguenti:

- cambiamento nei requirements
- perdita di dati
- dipendenza da servizi esterni
- affidabilità e compatibilità dei sensori
- comportamenti inaspettati dei clienti
- concorrenza
- sponsorizzare PowerEnJoy