PowerEnjoy

Riccardo Redaelli
Nicola Sosio
Maria Chiara Zaccardi

Politecnico di Milano

6 Marzo 2017

Introduzione

PowerEnjoy è una società di car sharing che offre soltanto macchine elettriche.
Permette inoltre agli utenti di noleggiare una macchina per brevi viaggi e invia notifiche agli operatori con le richieste di riparazione o di assistenza.
PowerEnjoy vuole incentivare il comportamente virtuoso degli utenti attraverso sconti.

Assunzioni 1/2

- Gli operatori ricevono le credenziali di accesso al momento della loro assunzione.
- 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 della loro assunzione.
- 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 della loro assunzione.
- 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 della loro assunzione.
- 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 può pagare.
- Il parcheggio è consentito solo all'interno di una Safe Area, fuori da essa il noleggio non può terminare.
- 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 soltant a carica completa.

Assunzioni 2/2

- L'account viene bloccato se l'utente effettua un noleggio per cui non può pagare.
- Il parcheggio è consentito solo all'interno di una Safe Area, fuori da essa il noleggio non può terminare.
- 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 soltant a carica completa.

Assunzioni 2/2

- L'account viene bloccato se l'utente effettua un noleggio per cui non può pagare.
- Il parcheggio è consentito solo all'interno di una Safe Area, fuori da essa il noleggio non può terminare.
- 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 soltant a carica completa.

Assunzioni 2/2

- L'account viene bloccato se l'utente effettua un noleggio per cui non può pagare.
- Il parcheggio è consentito solo all'interno di una Safe Area, fuori da essa il noleggio non può terminare.
- 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 soltant a carica completa.

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**: persone 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 delle corse.

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**: persone 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 delle corse.

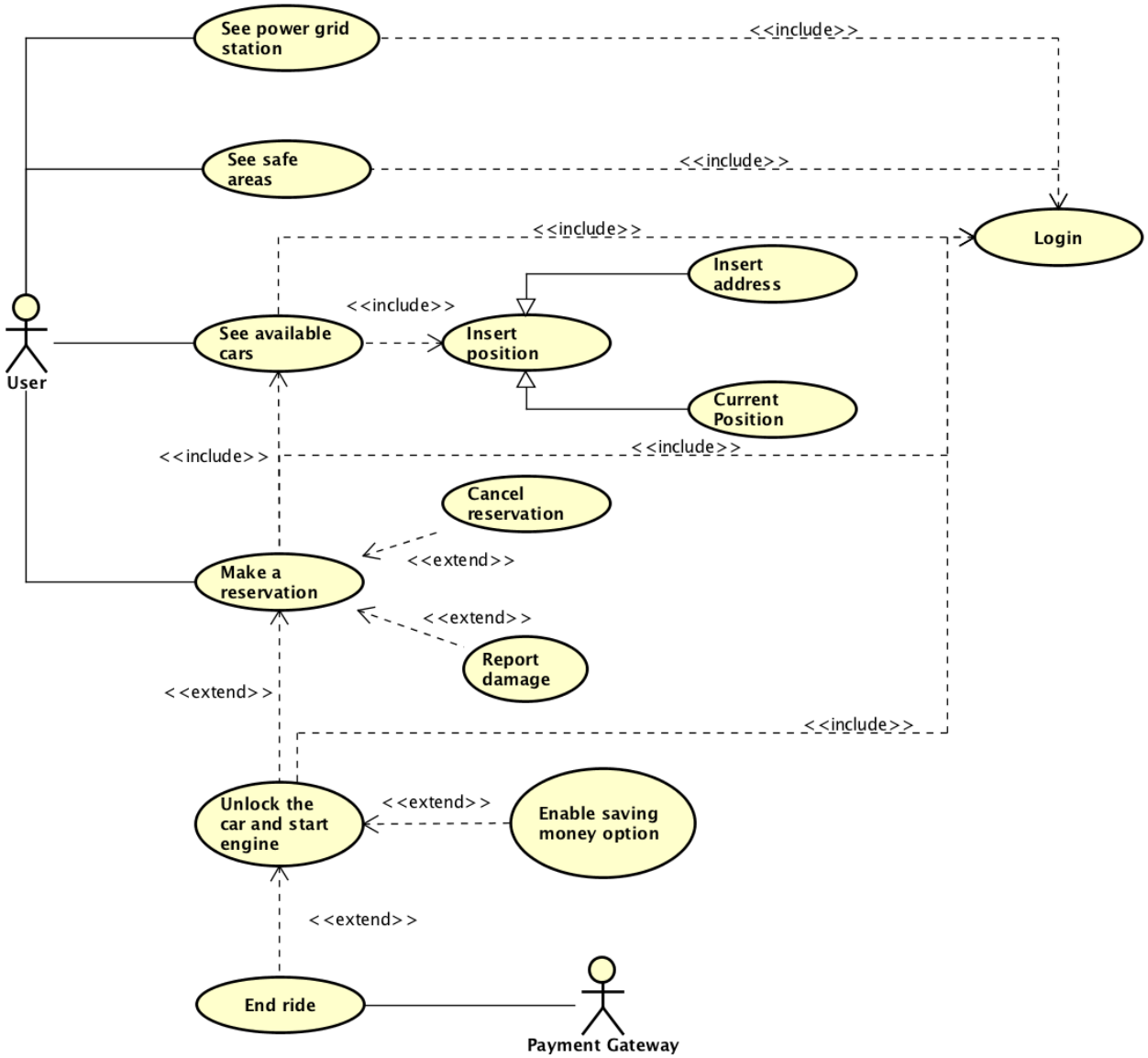
Attori

- **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**: persone 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 delle corse.

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**: persone 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 delle corse.

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>>
<<include>>
Unlock the car and start engine
<<extend>>
Enable saving money option
<<extend>>
End ride
Payment Gateway

Use case 2/2

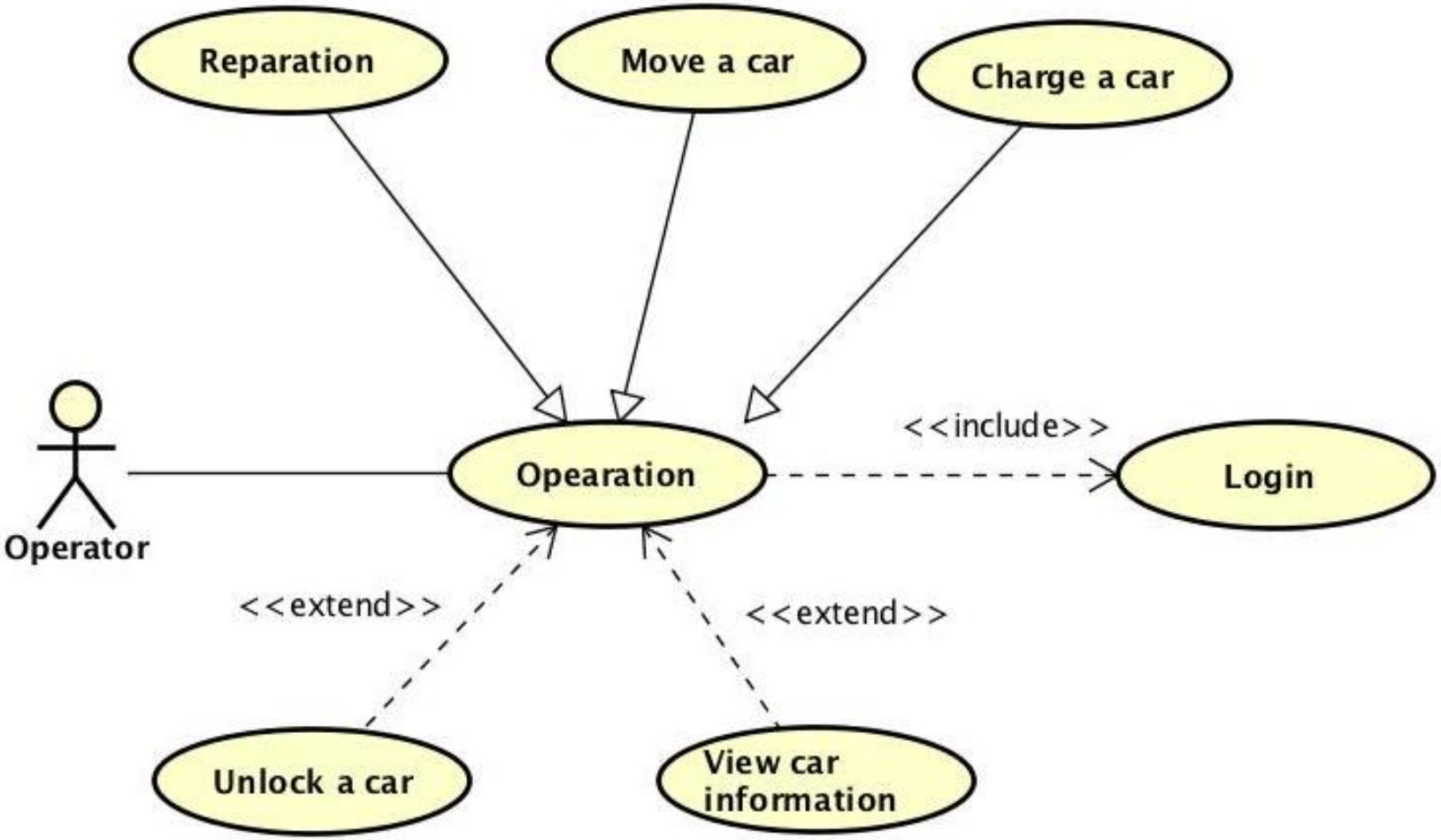
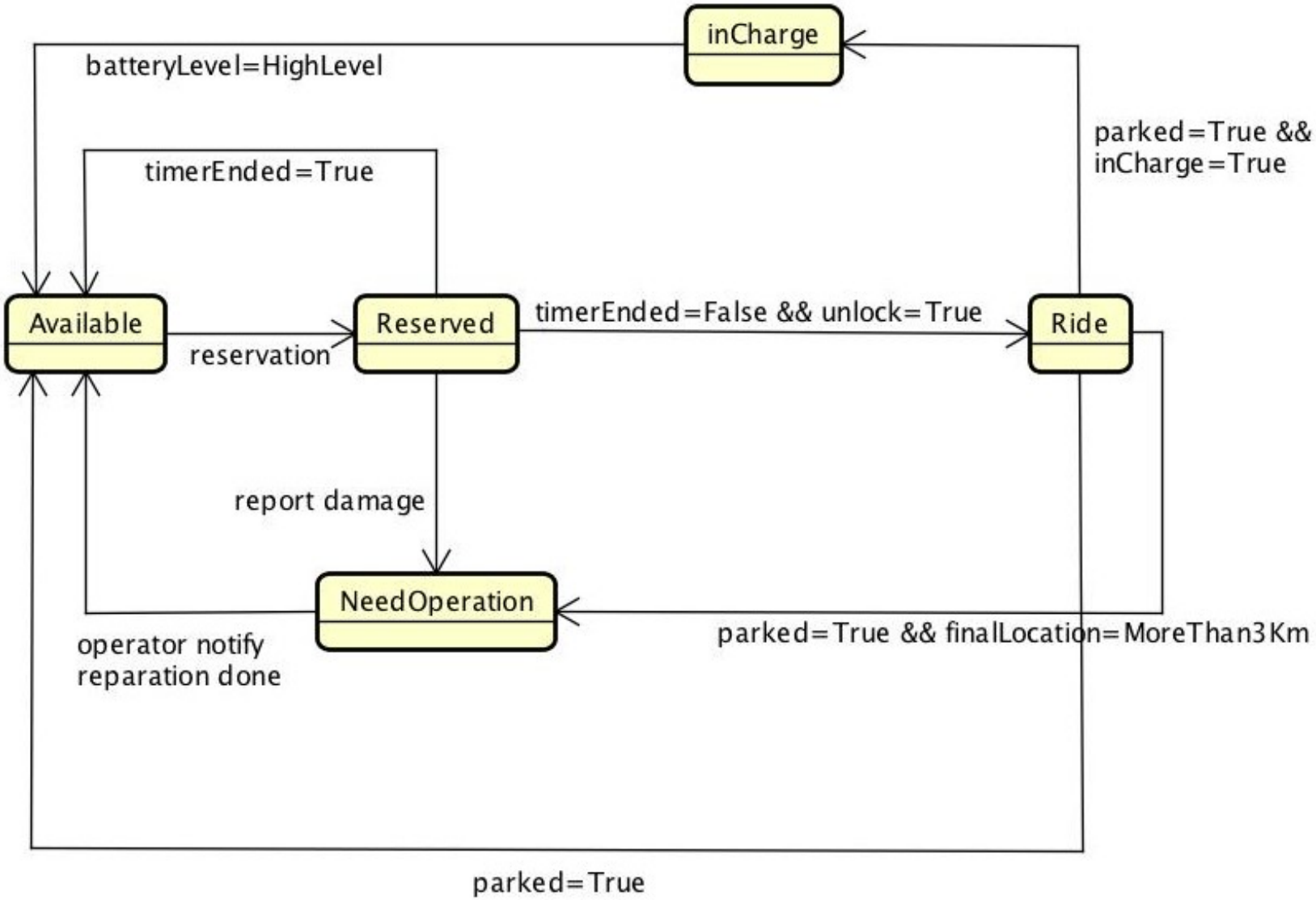

Reparation
Move a car
Charge a car
Operator
Opearation
<<include>>
Login
<<extend>>
<<extend>>
Unlock a car
View car information

Diagramma a stati



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

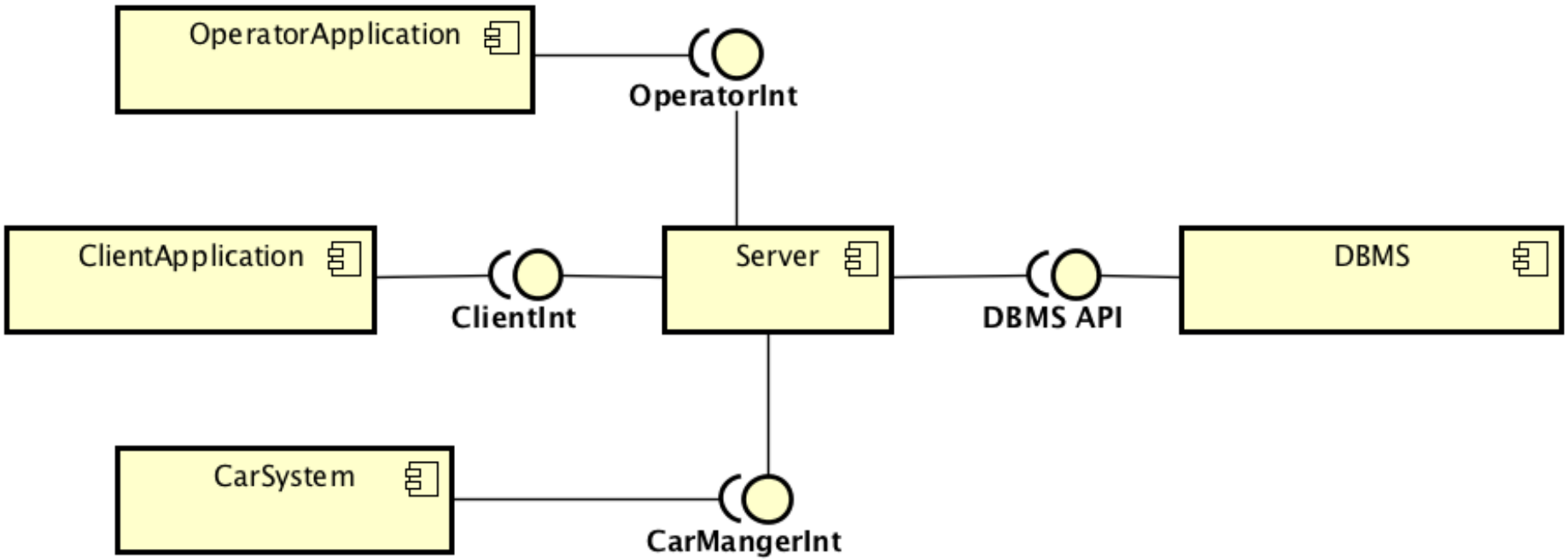
Architettura

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

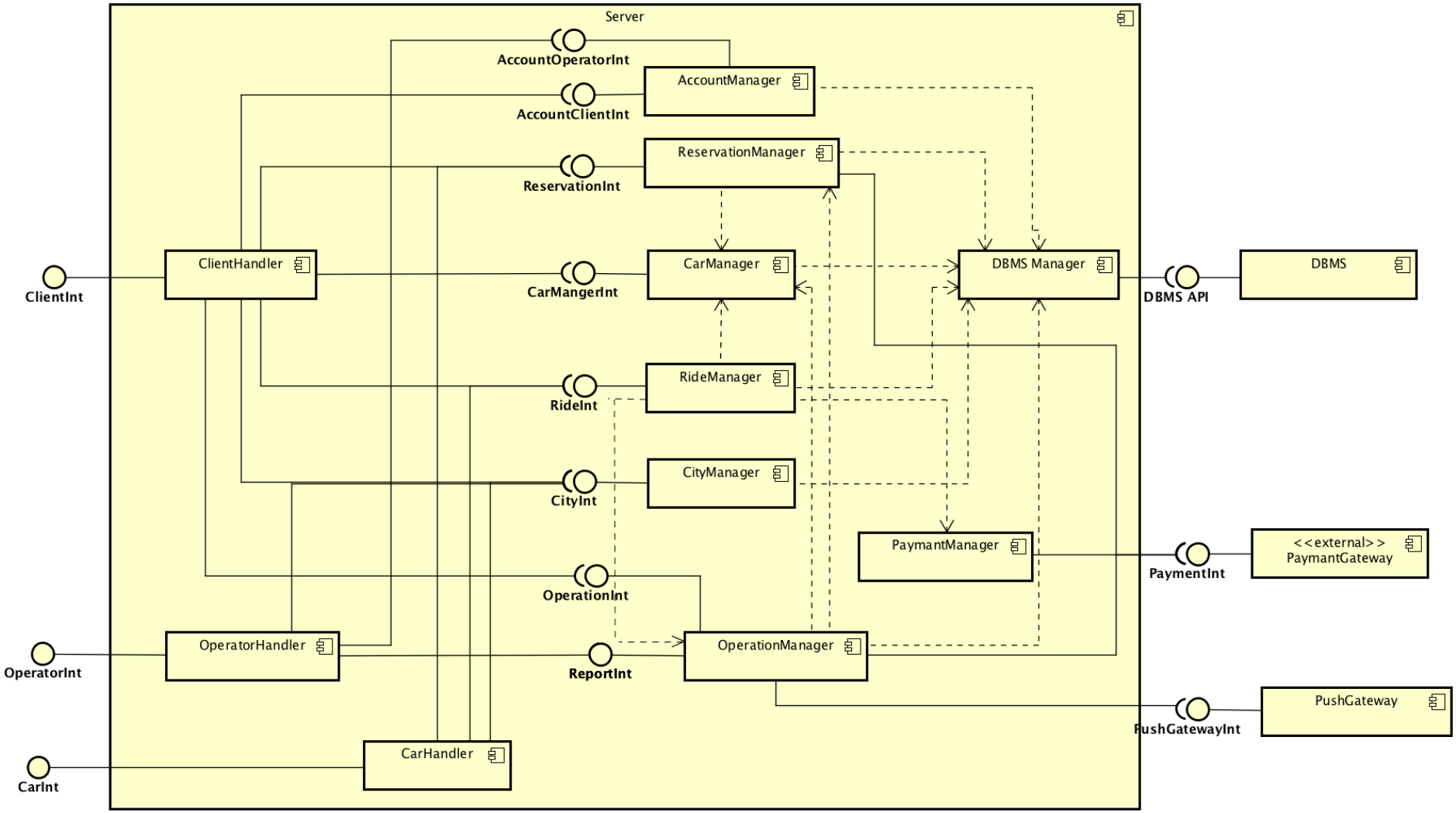
Architettura

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

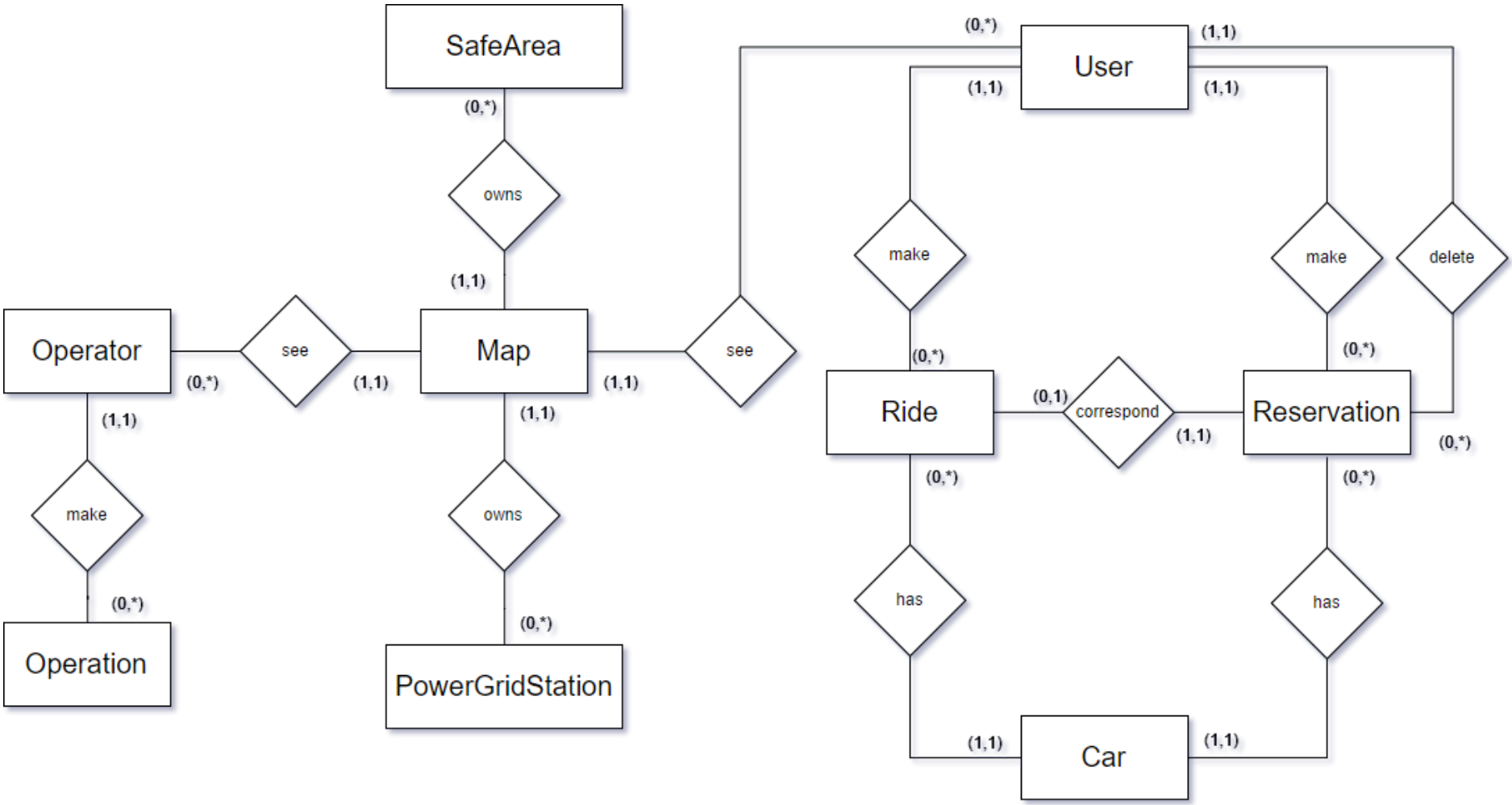
High level components



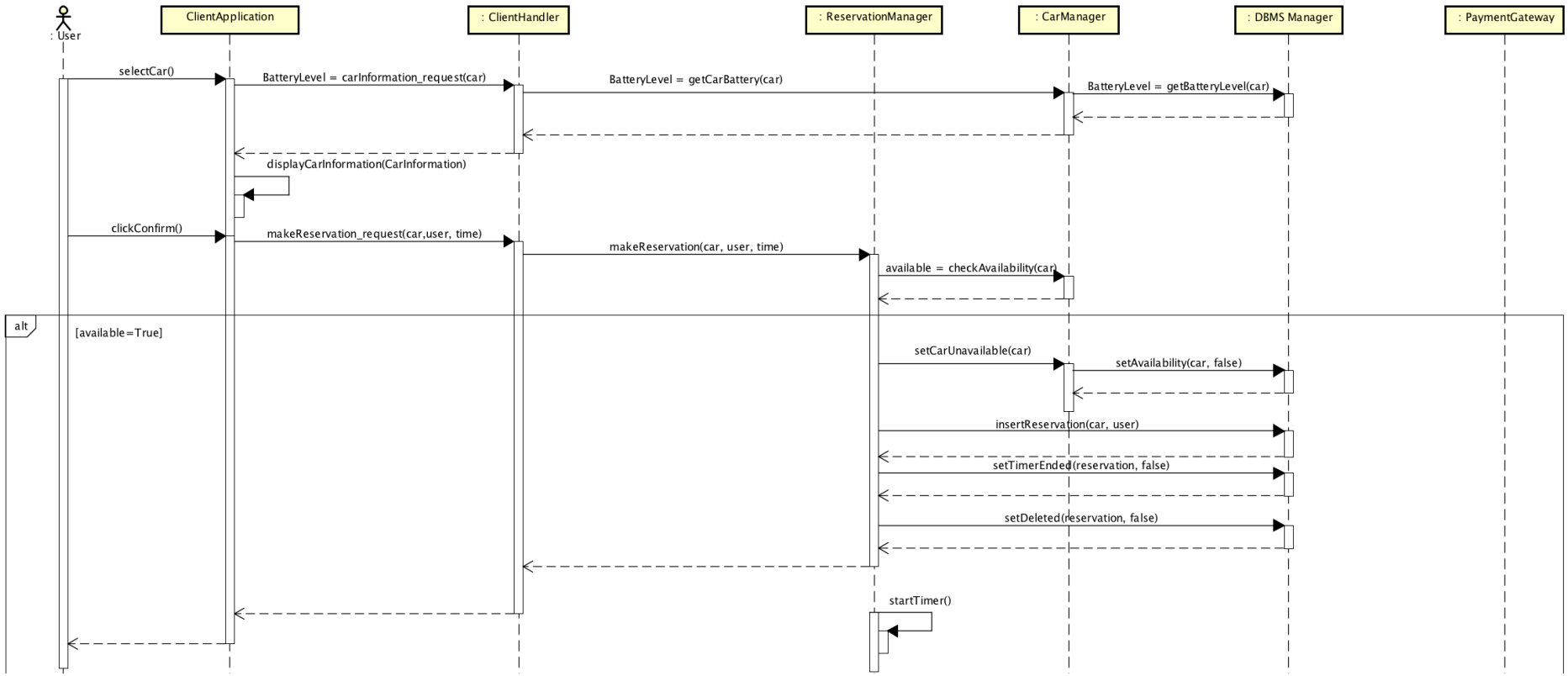
Component view



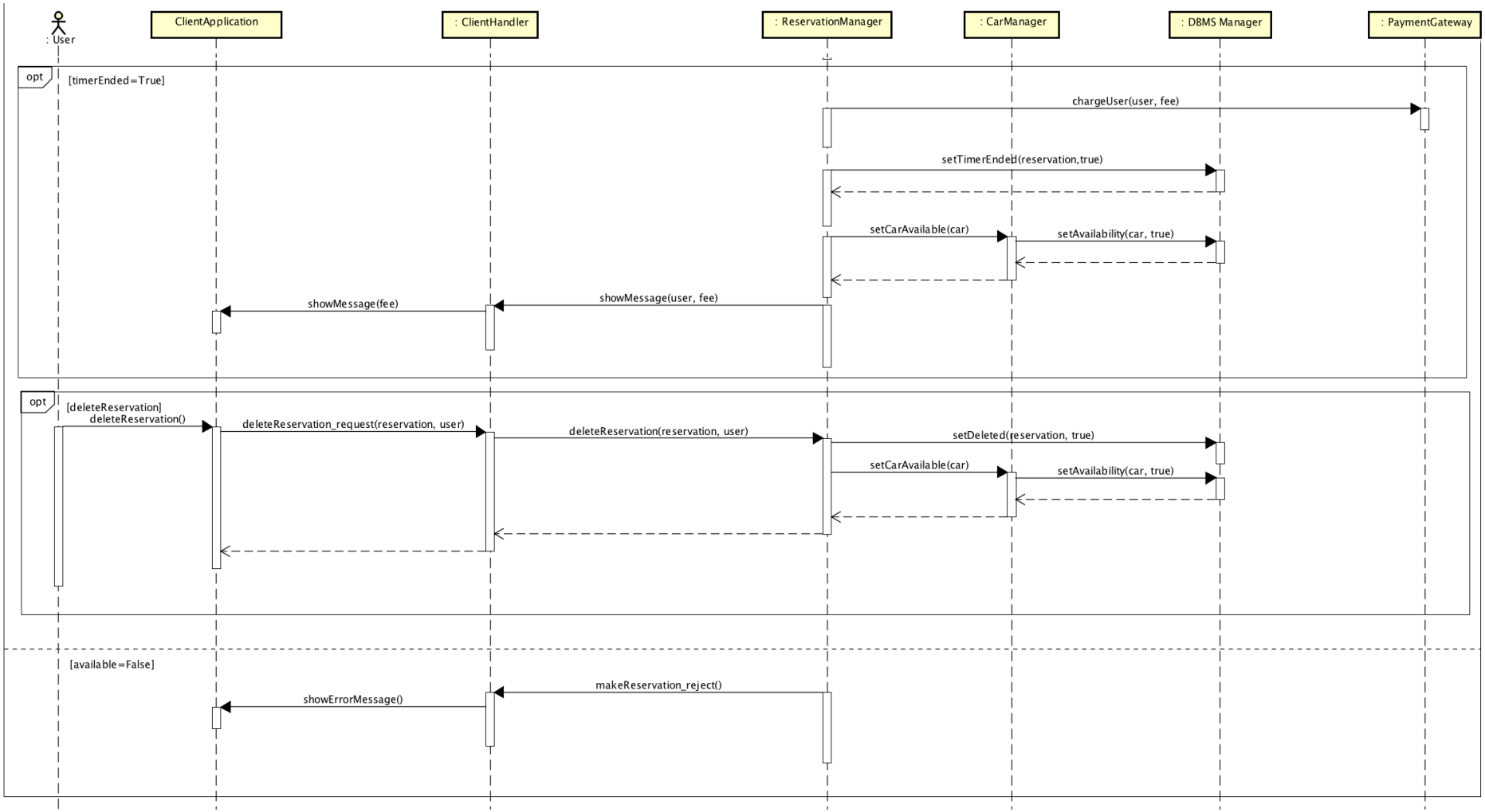
Struttura database



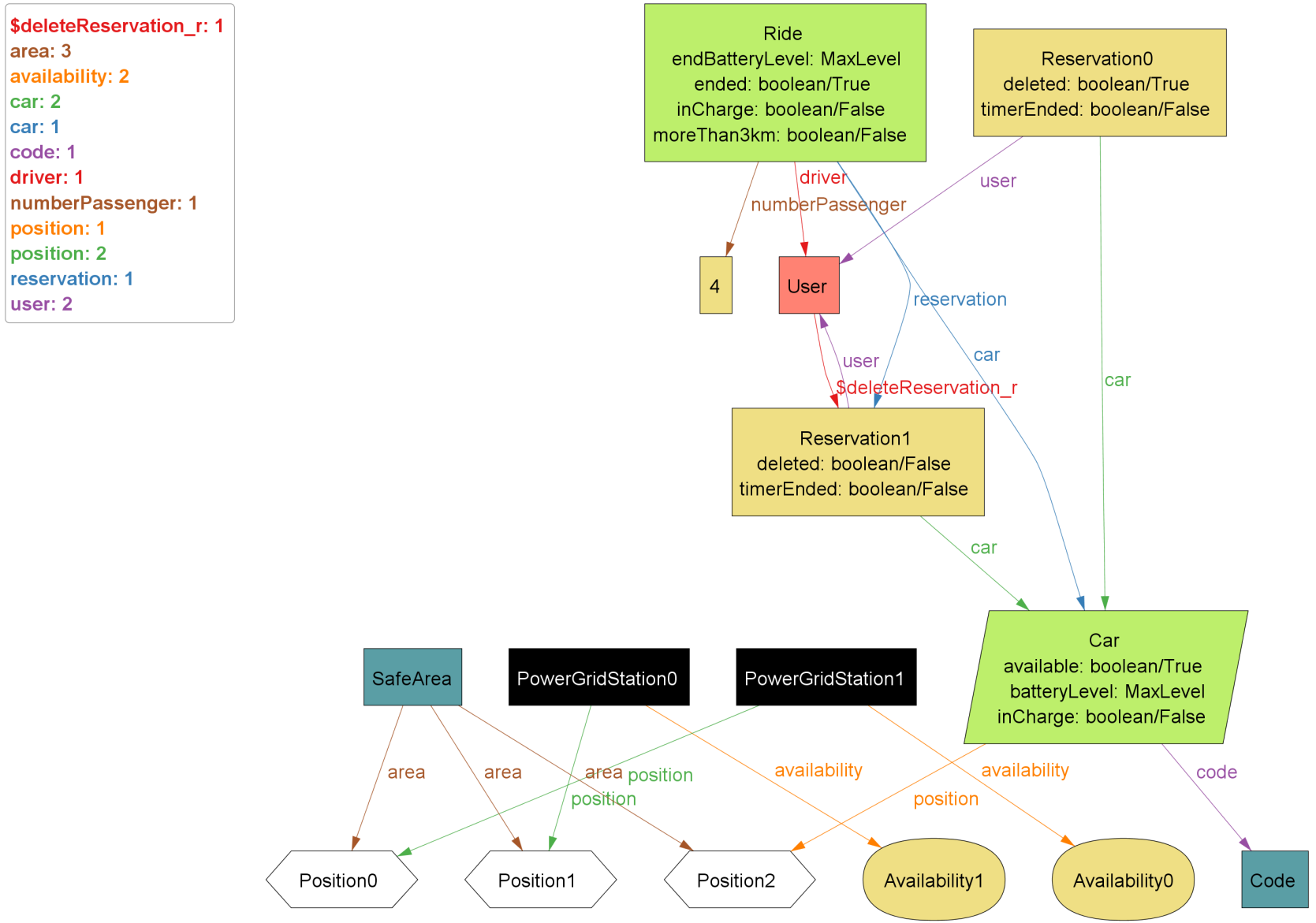
Sequence diagram - Prenotazione 1/2



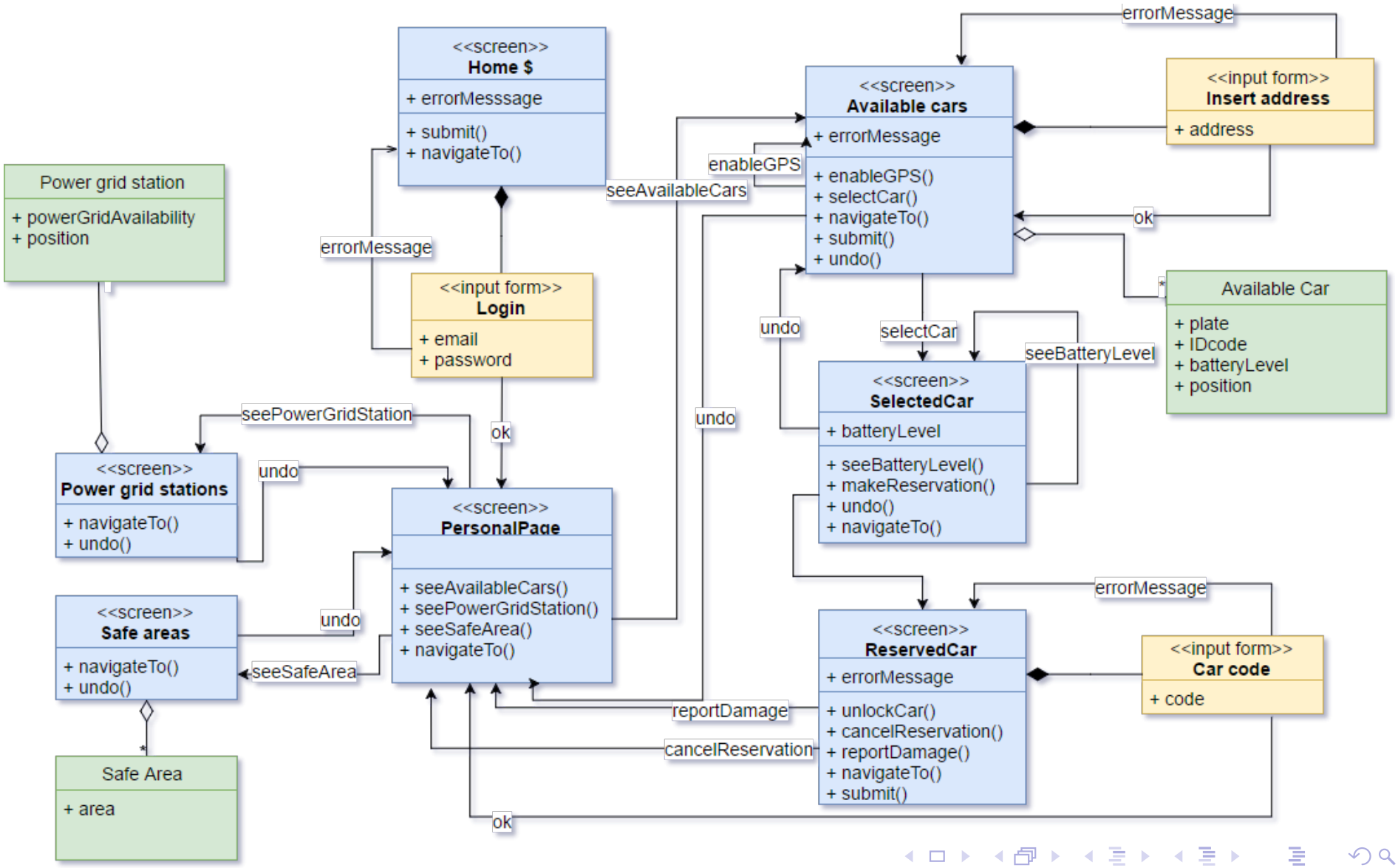
Sequence diagram - Prenotazione 2/2



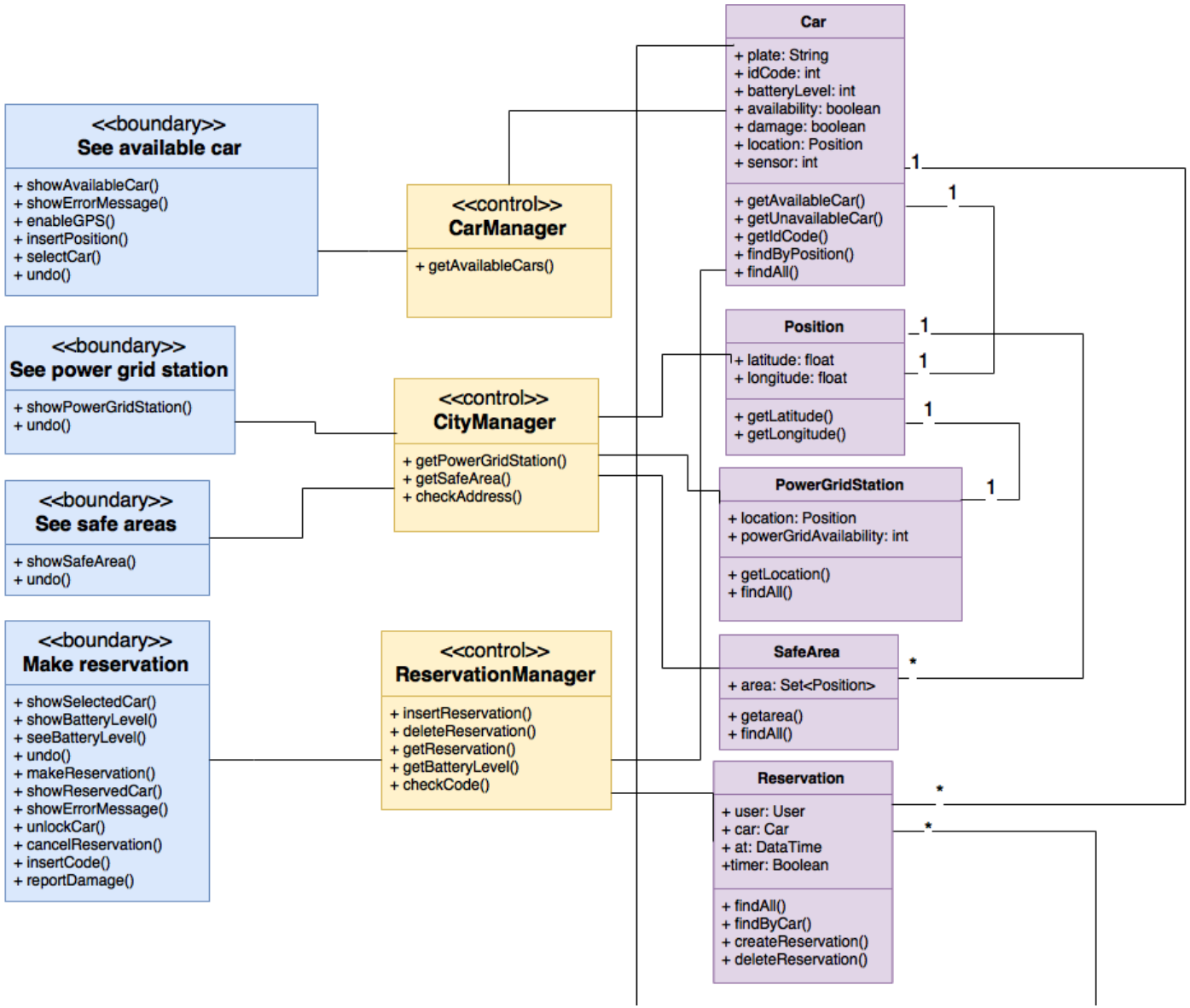
Alloy - Cancellazione prenotazione



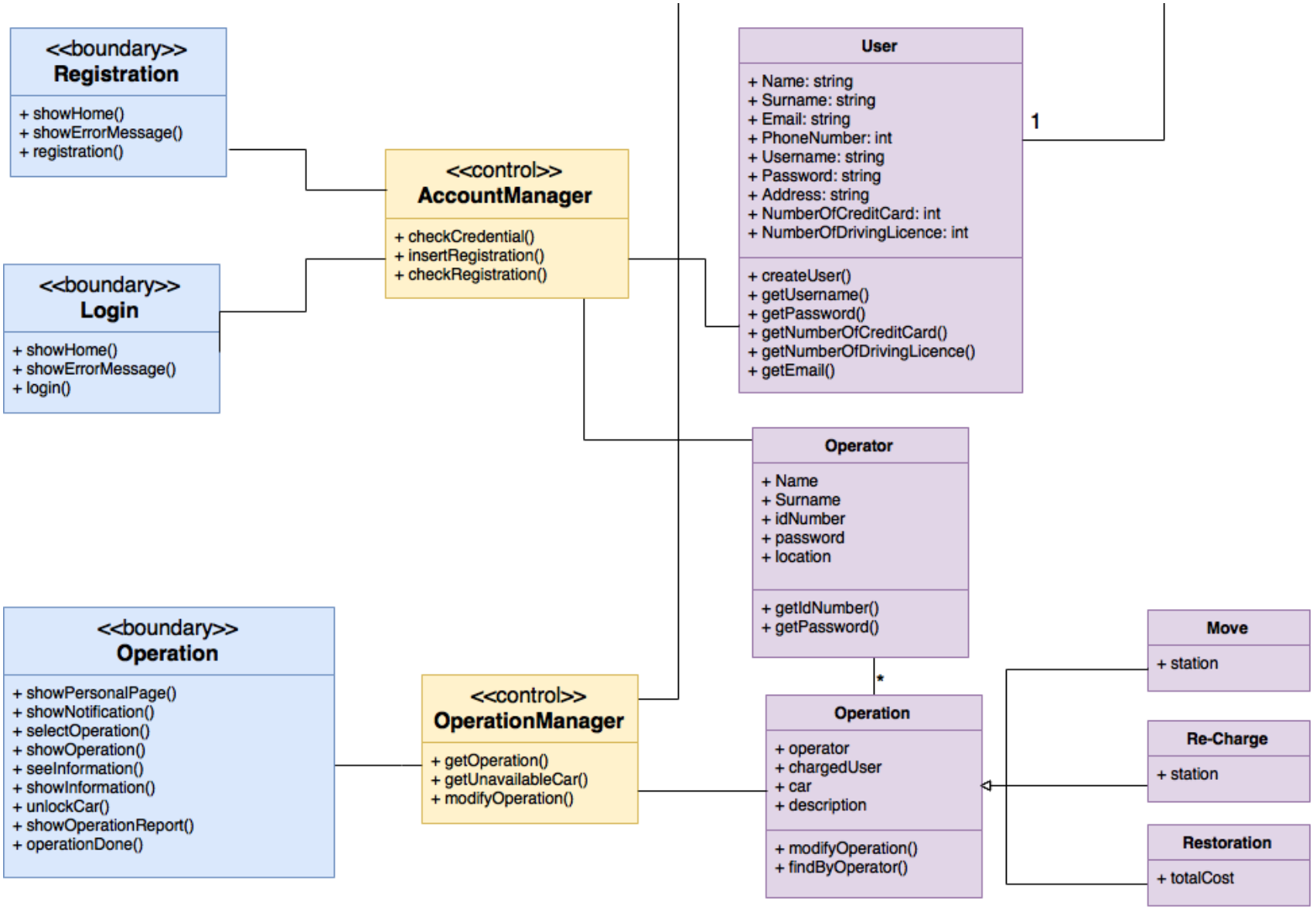
UX User


<<screen>>
Home $
+ errorMesssage
+ submit()
+ navigateTo()
<<screen>>
Available cars
+ errorMessage
+ enableGPS()
+ selectCar()
+ navigateTo()
+ submit()
+ undo()
<<input form>>
Insert address
+ address
errorMessage
enableGPS
seeAvailableCars
ok
Power grid station
+ powerGridAvailability
+ position
errorMessage
<<input form>>
Login
+ email
+ password
Available Car
+ plate
+ IDcode
+ batteryLevel
+ position
undo
selectCar
seeBatteryLevel
<<screen>>
SelectedCar
+ batteryLevel
+ seeBatteryLevel()
+ makeReservation()
+ undo()
+ navigateTo()
seePowerGridStation
ok
undo
<<screen>>
Power grid stations
+ navigateTo()
+ undo()
undo
<<screen>>
PersonalPage
+ seeAvailableCars()
+ seePowerGridStation()
+ seeSafeArea()
+ navigateTo()
errorMessage
<<screen>>
Safe areas
+ navigateTo()
+ undo()
undo
seeSafeArea
<<screen>>
ReservedCar
+ errorMessage
+ unlockCar()
+ cancelReservation()
+ reportDamage()
+ navigateTo()
+ submit()
<<input form>>
Car code
+ code
reportDamage
cancelReservation
Safe Area
+ area
ok

BCE User 1/2


<<boundary>>
See available car
+ showAvailableCar()
+ showErrorMessage()
+ enableGPS()
+ insertPosition()
+ selectCar()
+ undo()
<<control>>
CarManager
+ getAvailableCars()
<<boundary>>
See power grid station
+ showPowerGridStation()
+ undo()
<<control>>
CityManager
+ getPowerGridStation()
+ getSafeArea()
+ checkAddress()
<<boundary>>
See safe areas
+ showSafeArea()
+ undo()
<<boundary>>
Make reservation
+ showSelectedCar()
+ showBatteryLevel()
+ seeBatteryLevel()
+ undo()
+ makeReservation()
+ showReservedCar()
+ showErrorMessage()
+ unlockCar()
+ cancelReservation()
+ insertCode()
+ reportDamage()
<<control>>
ReservationManager
+ insertReservation()
+ deleteReservation()
+ getReservation()
+ getBatteryLevel()
+ checkCode()
Car
+ plate: String
+ idCode: int
+ batteryLevel: int
+ availability: boolean
+ damage: boolean
+ location: Position
+ sensor: int
+ getAvailableCar()
+ getUnavailableCar()
+ getIdCode()
+ findByPosition()
+ findAll()
Position
+ latitude: float
+ longitude: float
+ getLatitude()
+ getLongitude()
PowerGridStation
+ location: Position
+ powerGridAvailability: int
+ getLocation()
+ findAll()
SafeArea
+ area: Set<Position>
+ getarea()
+ findAll()
Reservation
+ user: User
+ car: Car
+ at: DataTime
+timer: Boolean
+ findAll()
+ findByCar()
+ createReservation()
+ deleteReservation()
1
1
1
1
1
1
*
*
*

BCE User 2/2


<<boundary>>
Registration
+ showHome()
+ showErrorMessage()
+ registration()
<<boundary>>
Login
+ showHome()
+ showErrorMessage()
+ login()
<<boundary>>
Operation
+ showPersonalPage()
+ showNotification()
+ selectOperation()
+ showOperation()
+ seeInformation()
+ showInformation()
+ unlockCar()
+ showOperationReport()
+ operationDone()
<<control>>
AccountManager
+ checkCredential()
+ insertRegistration()
+ checkRegistration()
<<control>>
OperationManager
+ getOperation()
+ getUnavailableCar()
+ modifyOperation()
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()
1
Operator
+ Name
+ Surname
+ idNumber
+ password
+ location
+ getIdNumber()
+ getPassword()
*
Operation
+ operator
+ chargedUser
+ car
+ description
+ modifyOperation()
+ findByOperator()
Move
+ station
Re-Charge
+ station
Restoration
+ totalCost

Prerequisiti

- **RASD** e **DD** devono essere stati precedentemente redatti.
- Tutte le classi di ogni component devono essere correttamente **documentate**.
- Ogni classe deve essere stata testata attraverso i **test d'unità**.

Prerequisiti

- **RASD** e **DD** devono essere stati precedentemente redatti.
- Tutte le classi di ogni component devono essere correttamente **documentate**.
- Ogni classe deve essere stata testata attraverso i **test d'unità**.

Prerequisiti

- **RASD** e **DD** devono essere stati precedentemente redatti.
- Tutte le classi di ogni component devono essere correttamente **documentate**.
- Ogni classe deve essere stata testata attraverso i **test d'unità**.

Strategia di Integrazione

- Bottom-up approach
  - I componenti vengono testati a partire da quelli senza dipendenze
  - Gli stubs non sono necessari
  - Driver temporanei

Strategia di Integrazione

- Bottom-up approach
  - I componenti vengono testati a partire da quelli senza dipendenze
  - Gli stubs non sono necessari
  - Driver temporanei

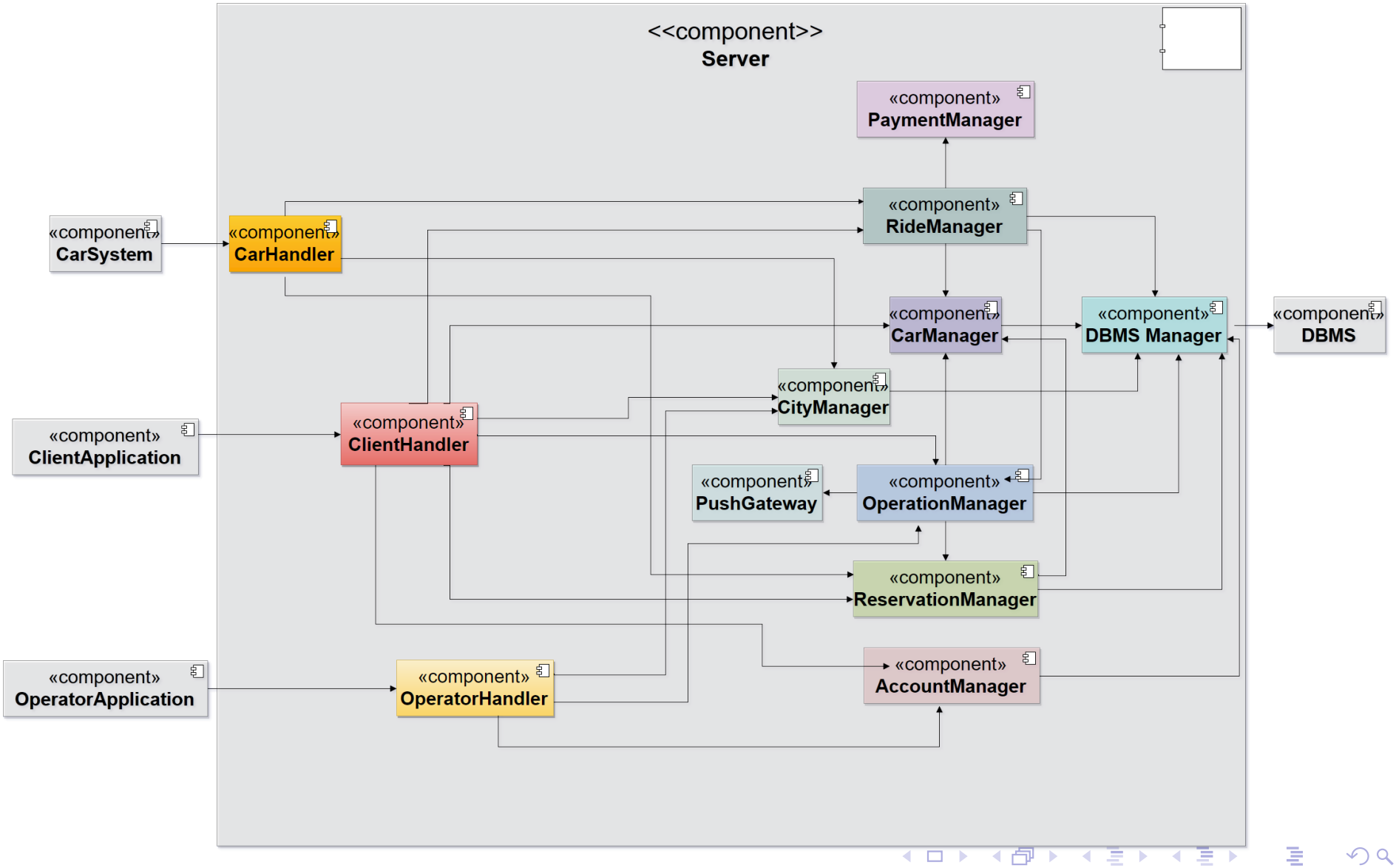
Strategia di Integrazione

- Bottom-up approach
  - I componenti vengono testati a partire da quelli senza dipendenze
  - Gli stubs non sono necessari
  - Driver temporanei

Strategia di Integrazione

- Bottom-up approach
  - I componenti vengono testati a partire da quelli senza dipendenze
  - Gli stubs non sono necessari
  - Driver temporanei

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

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.

External Input

External Input	Complexity	FPs
Registration	Simple	3
Login	Simple	3
Make a reservation	Medium	4
Cancel a reservation	Medium	4
Enable money saving option	Complex	6
Report damage	Complex	6
Plug the car	Simple	3
Unlcok the car (insert the code)	Simple	3
Login	Simple	3
Accept an operation	Simple	3
Report operations informations	Simple	3
Total		41

External Output

External Output	**Complexity**	**FPs**
Notify an operator	Simple	4
Provide a password	Simple	4
Apply possible fee or discount	Medium	7
Show the map	Medium	7
Show the total amount	Simple	4
Total		26

External Inquiry

External Inquiry	Complexity	FPs
Select a car	Simple	3
See reservation informations	Simple	3
See available car (by GPS)	Simple	3
See available car (by address)	Simple	3
See power grid station and their position	Simple	3
Availability of each Power Grid Station in real time	Simple	3
See Safe Area	Simple	3
Amount of the cost of the ride	Simple	3
See unavailable cars that need an operation	Simple	3
Total		27

Internal Logic File

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

External Logic File

External Logic File	**Complexity**	**FPs**
GPS signal	Complex	10
Reverse geocoding	Complex	10
Map data retrieval	Complex	10
Total		30

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

$$SLOC = 46 * 201 = 9246$$

Scale Drivers

Scale Factors	Very Low	Low	Nominal	High	Very High	Extra High
Prec	thoroughly unprecedented	largely unprecedented	somewhat unprecedented	generally familiar 2.48	largely familiar 1.24	throughly familiar
SF_j	6.20	4.96	3.72	2.48	1.24	0.00
Flex	rigorous	occasional relaxation	some relaxation	general conformity	some conformity	general goals
SF_j	5.07	4.05	3.04	2.03	1.01	0.00
Resl	little (20%)	some (40%)	often (60%)	generally (75%)	mostly (90%)	full (100%)
SF_j	7.07	5.65	4.24	2.83	1.41	0.00
Team	very difficult interactions	some difficult interactions	basically cooperative interactions	largely cooperative	highly cooperative	seamless interactions
SF_j	5.48	4.38	3.29	2.19	1.10	0.00
Pmat	Level Lower 1	Level Upper 1	Level 2	Level 3	Level 4	Level 5
SF_j	7.80	6.24	4.68	3.12	1.56	0.00

Cost Drivers

Feature	Factor	Value
Required Software Reliability	Low	0.92
Database size	Nominal	1.00
Product complexity	Nominal	1.00
Required Reusability	Nominal	1.00
Documentation match to life-cycle needs	Nominal	1.00
Execution Time Constraint	High	1.11
Main storage constrain	Hominal	1.00
Platform volatility	Low	0.87
Analyst capability	High	0.85
Programmer capability	Nominal	1.00
Personnel continuity	High	0.81
Application Experience	Very Low	1.22
Platform Experience	Very Low	1.19
Language and Tool Experience	Low	1.09
Usage of Software Tools	High	0.90

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

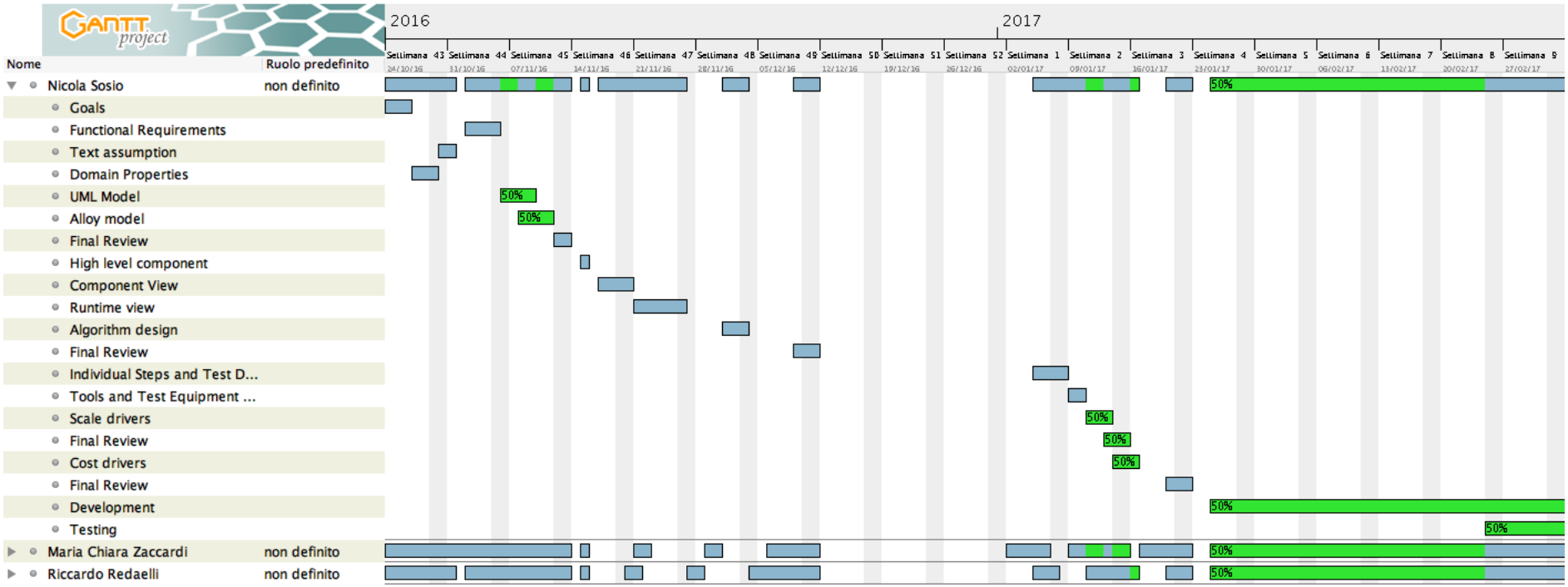
Schedule

$$TDEV = [C * PM^F] * \frac{SCED\%}{100} = 3.67 * 27^{0.3157} * \frac{130}{100} \simeq 14 months$$
$$F = D + 0.2 * (E - B) = 0.28 + 0.2 * (1.86 - 0.91) = 0.3157$$

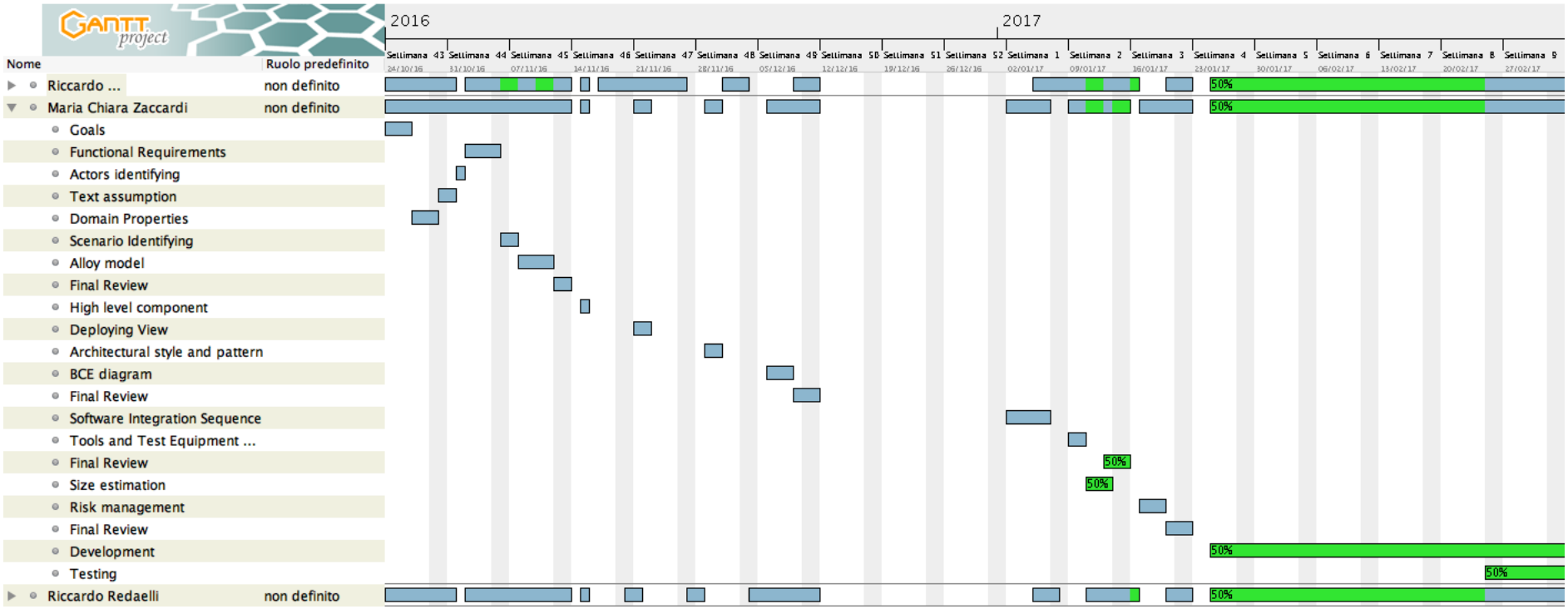
dove:

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

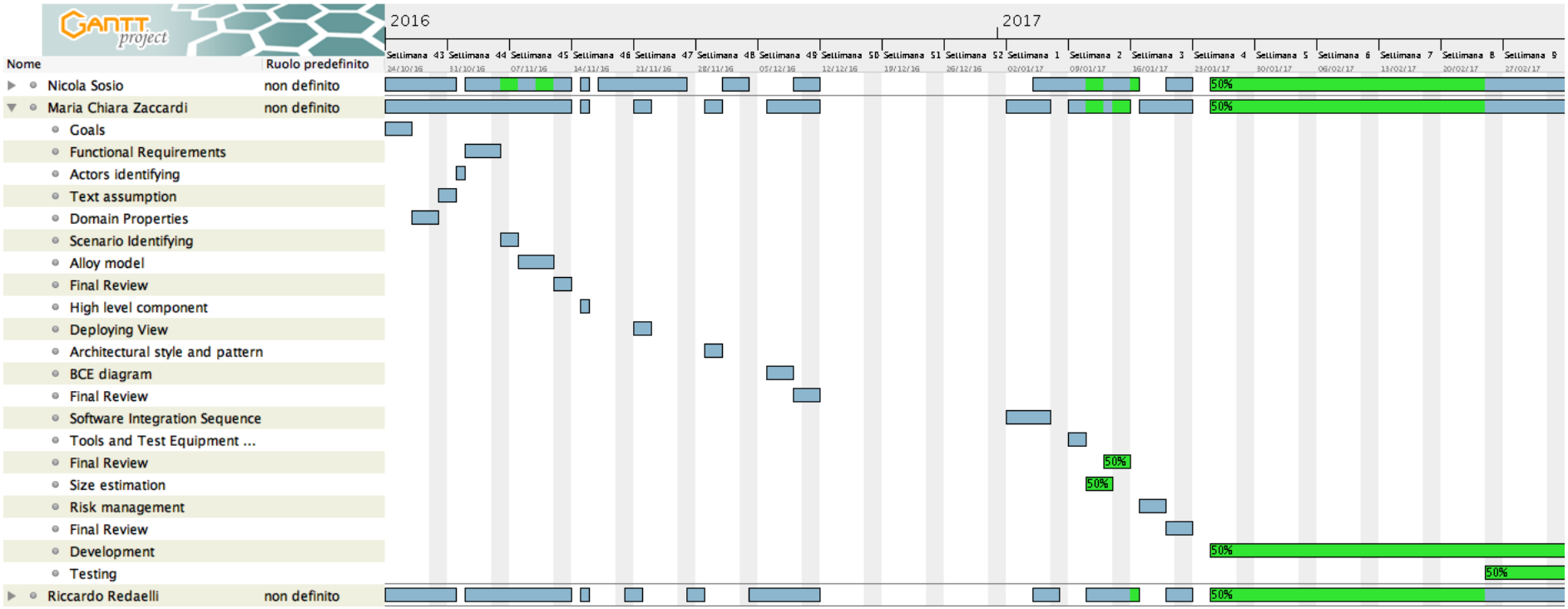
Allocazione risorse 1/3



Allocazione risorse 2/3



Allocazione risorse 3/3



Rischi

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

Rischi

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

Rischi

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

Rischi

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

Rischi

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

Rischi

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

Rischi

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