myTaxiService

RASD

Luca Nanni

Giacomo Servadei

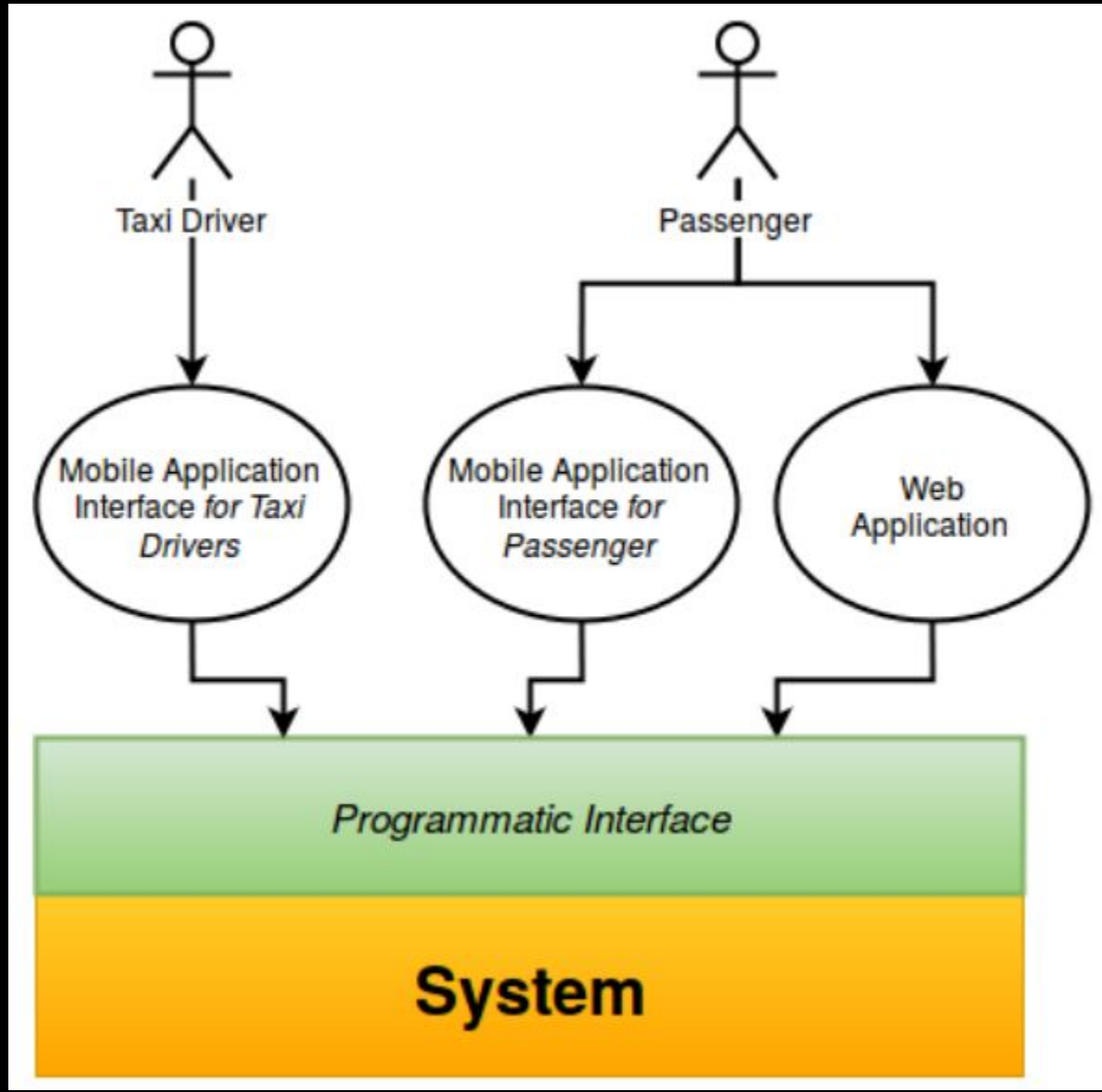
Table of contents

Introduction

- **Purpose**
  - The intended audience of this paper is:
    - The project manager
    - The client (Government of the city)
    - The designers and developers of the application
    - The testing team
    - The end user

Introduction

- The system is designed to interact with 2 user classes:
  - Passengers
    - Mobile Application & Web Application
  - Taxi Drivers
    - Mobile Application
- Moreover, it is required to implement a programmatic interface for improving future development



Introduction

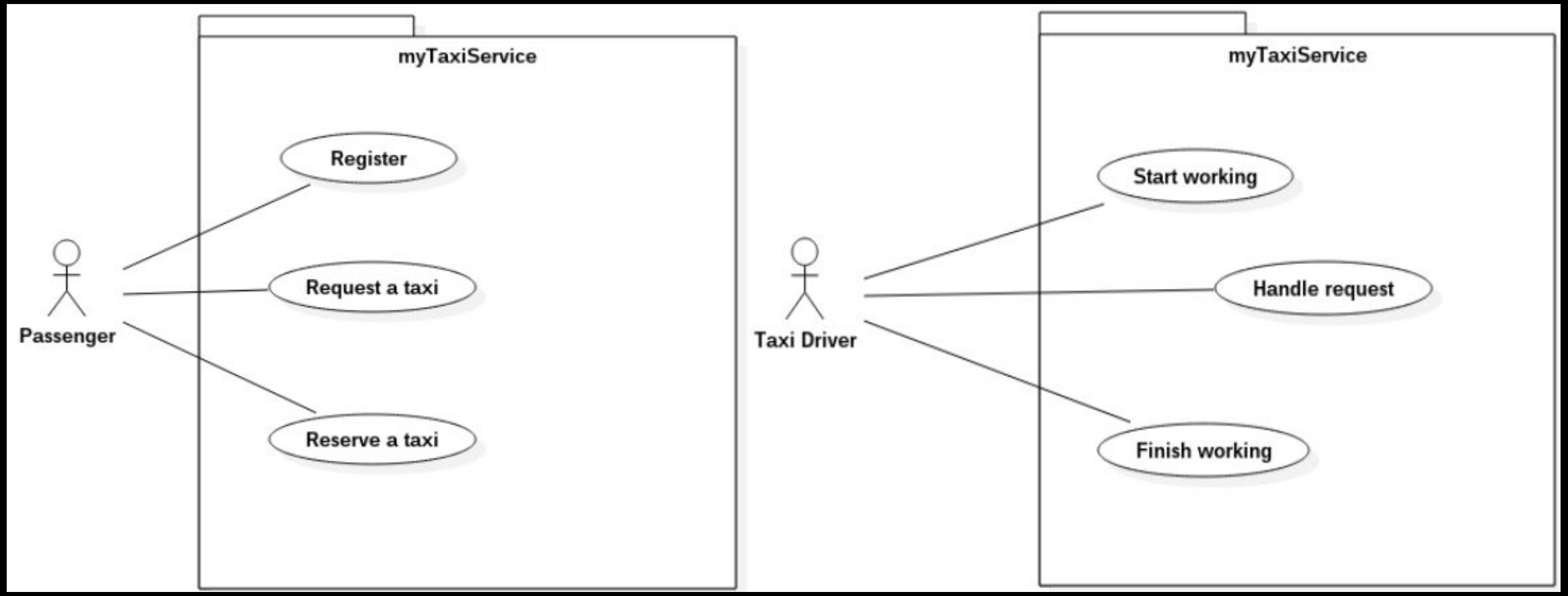
Glossary

System	It is the application we have to design. It is constituted by different interfaces: two for the Passengers, one for the Taxi Drivers and one for the future extension of the application (programmatic interface).
Taxi driver	One of the user of the system. They are the people which task is to drive the taxi. They use the application to communicate their availability or not and to be assigned to the taxi queues.
Taxi queue	It is an abstract queue of the available taxis in a taxi zone of the city. It has an order determined by the time in which each taxi driver communicate its availability to the system or by the decision of the taxi driver to accept a request or not
Taxi service	It is a service offered by the government of the large city which enable every person to use the offered taxis. It can be of different nature as a reservation service or a request.
Taxi zone	The city is divided in taxi zones, each one having its taxi queue. The division is provided in order to distribute the taxi availability in all the city territory
Web application	Is one of the interfaces that the passengers can use to interact with the system. To use it the passenger must use a web browser.

Term	Description
Available	Status of a Taxi Driver in which he can receive requests
Busy	Status of a Taxi Driver in which he is serving a Passenger's request
Government of the city	It is the client for which we are working. It desires an application for the improvement and simplification of the taxi service.
City	The ambient in which the taxi drivers and the passengers interact. It is divided in taxi zones.
GPS	Technology which manage to get in every moment the position of a vehicle
Meeting point/location	It is the location inside the city at which the passenger and the taxi driver will meet. It is set by the Passenger during the reservation procedure
Meeting time	It is the time at which the passenger and the taxi driver will meet. It is set by the Passenger during the reservation and request procedures
Mobile application	It is one of the interface that passengers and taxi drivers can use to interact with the system. To use it they must have it installed in their smart phone.
Not available	Status of a Taxi Driver in which he cannot receive requests
Passenger	One of the user of the system. He can apply for a taxi service: he can make a request or a reservation
Programmatic interface	It is a software interface to be used by developers to modify and extend the actual software. It is useful for the extension of the application with additional taxi services
Queue management	It is the part of the system that takes care of the organization of the taxi drivers inside the taxi queue of the correspondent taxi zone
Request	It is the action carried out by the passenger when he needs to use the taxi service. It represents an immediate need of the passenger
Reservation	It is the action carried out by the passenger when he needs to use the taxi service in the future. It consists in the specification of the origin and the destination of the taxi ride, reserved at a desired time. The passenger can reserve a journey only if the specified time is at least two hours after the time of reservation.

Overall Description

- **Product functions**



Overall Description

- **Goals**
  - A **Passenger**
    - must be able to register/login to the service
    - must be able to access the application either from the web application or the mobile application
    - must be able to request a taxi ride from a location he decides
    - must be able to reserve a taxi ride from an origin, to a destination at a specific time and date
    - must be able to cancel a reservation that he made
    - must receive, if a taxi is available for a request, a notification stating the arrival time and the taxi identification number
    - if he has reserved a taxi correctly for a certain meeting time, he must be notified before that time with a notification saying the arrival time of the taxi and it identification number

Overall Description

- **Goals**
  - A **Taxi Driver**:
    - must be able to set himself as available (start working)
    - must be able to set himself as not available (stop working)
    - must receive requests (both immediate and reservations) only if he is available
    - must be able to choose if he wants to accept an incoming request or not
    - must receive a notification if he is the first taxi in the queue corresponding to his actual
      location and if a passenger made an request in that zone
    - when available, must be set in the correct waiting queue
    - must receive the location of the passenger who made the request, if he accepted it

Overall Description

•Assumptions

- Passenger side:
  - Once a passenger made a request and it has been accepted by a taxi, he's going to wait for it until it arrives and he will take the ride
  - The Passenger, once ended the ride, will pay the Taxi Driver directly. There is no support for an in-app payment system
  - Passengers can't access the mobile application used by the Taxi Drivers
  - **Passengers, once submitted a request, will never try to cancel it. ☹**
  - The Passengers will make requests and reservations only from a location inside the city only to a location inside the city

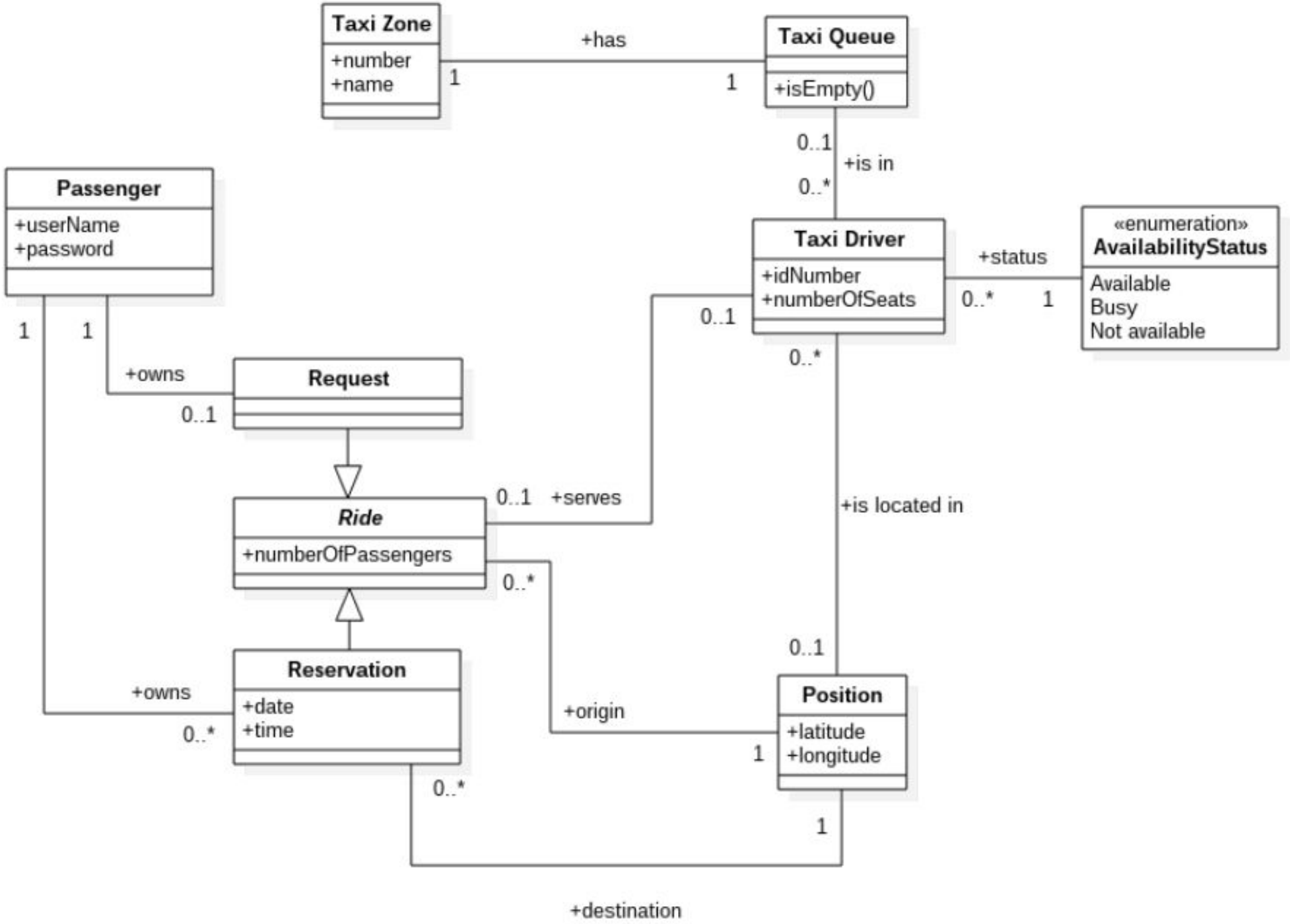
Overall Description

•Assumptions

- Taxi Driver side:
  - Each smart phone used by the Taxi drivers has a GPS system installed and the application has the privileges to access it
  - The taxi drivers will set their availability only when they can accept new passengers
  - The Taxi drivers will register to the system with a specific request to the Government of the city. There is no support for taxi driver registration in the system.
  - Taxi Drivers, once accepted a request, will always reach the meeting point
  - There is absolutely no overlapping between taxi zones
  - **Each taxi driver has only one taxi: this means that each taxi driver has a fixed identification number for his taxi**

Overall Description

Entities Involved


Taxi Zone
+number
+name
+has
1
1
Taxi Queue
+isEmpty()
0..1
+is in
0..*
Passenger
+userName
+password
Taxi Driver
+idNumber
+numberOfSeats
+status
0..*
1
«enumeration»
AvailabilityStatus
Available
Busy
Not available
0..1
0..*
1
1
+owns
0..1
Request
Ride
+numberOfPassengers
0..1
+serves
+is located in
0..*
Reservation
+date
+time
+owns
0..*
+origin
Position
+latitude
+longitude
1
0..1
0..*
1
+destination

Overall Description

- **Future Implementations**:
  - The system will be able to suggest the taxi driver in which taxi zone they should go in order to have better chance to receive requests. These suggestions will be computed based on the statistics regarding the number of requests per hour in each zone
  - The system could also manage the payment of the ride through an in-app interface
  - The system will have the possibility to handle the cancellation of the request after its submission to the system
  - The system will allow taxi drivers to accept reservations in advance. In this way, it is
    guaranteed that a reservation will be served at the right time, without delays.

Specific Requirements

Use case

Passenger registration

Use case	**Register**
Actors	Passenger
Goals	A passenger must be able to register to the service
Enter condition	None
Event flow	1. The passenger goes to the web application page 2. The passenger clicks on the sign up button 3. The passenger fills the form with username and password desired 4. The passenger clicks the submit button 5. If the username is already present in the system or there are missing data (a) Notify the Passenger of the error (b) Go back to Event flow 3 6. The system retrieves the data and stores them 7. The system notifies that the registration has been correctly done
Exceptions	If the user, during the insertion of the registration data (username + password) decides to abort the procedure by clicking on the proper button 1. The system notifies the passenger of the loss of the inserted data 2. The system abort the procedure
Exit condition	The passenger is correctly registered to the service

Use case

Request of a taxi (1)

Use case	**Request a taxi**
Actors	Passenger, Taxi Driver
Goals	A passenger must be able to request a taxi ride from a location he decides
Enter condition	• The Passenger must already be registered to the service • The Taxi driver must be registered to the service • The Passenger must already have opened the application (mobile or web) and logged in • The Taxi driver must be available

Event flow	1. The Passenger goes to the section for requesting a taxi ride
	2. The Passenger fills the form with:
	  • Meeting location
	  • Number of passengers
	3. The Passenger submit the request to the system
	4. If there are missing data or the specified location is not valid or the number of passengers is greater than 3:
	  (a) The system notifies the error to the Passenger
	  (b) Go back to Event flow 2

Use case

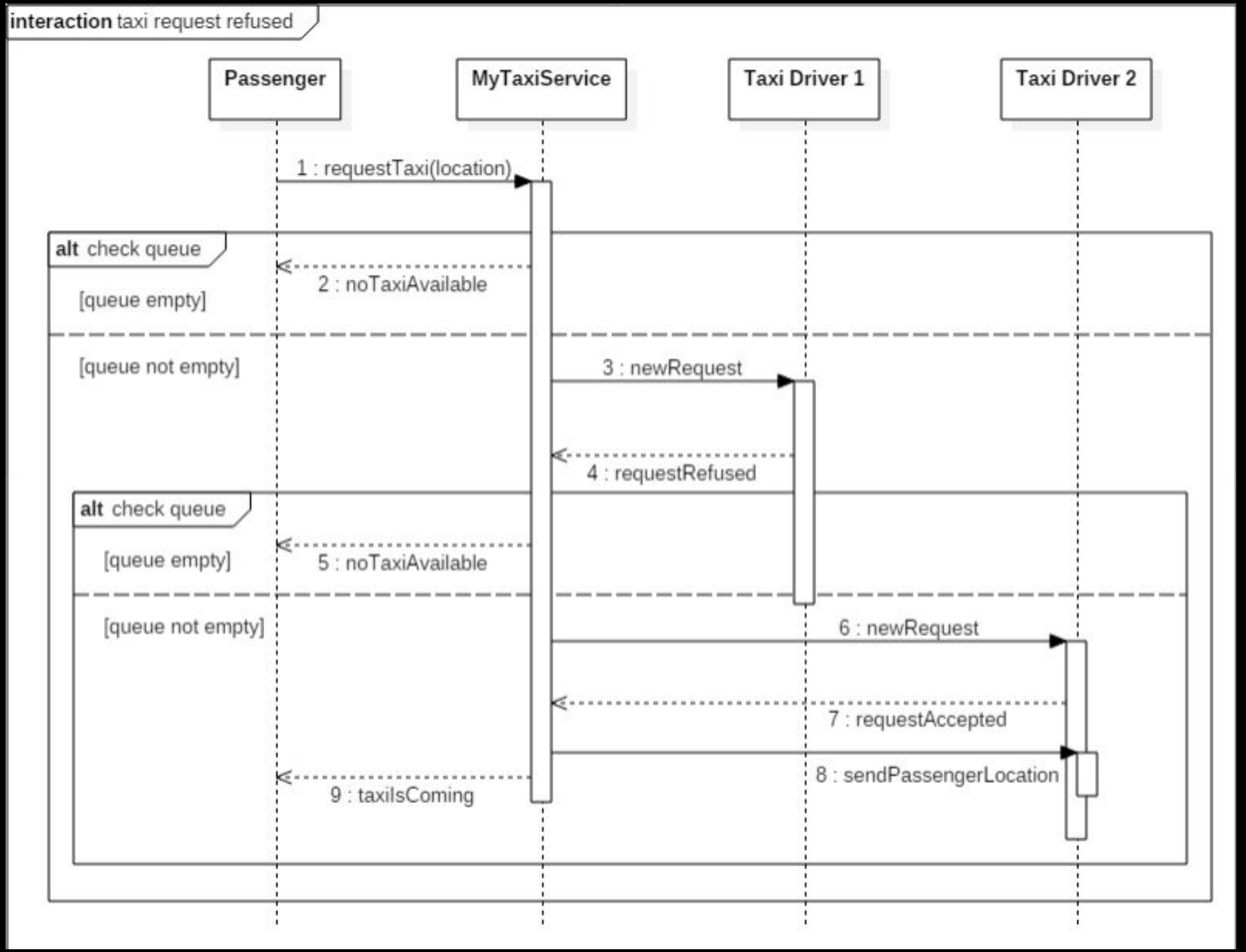
Request of a taxi (2)

	5. The request is stored in the system 6. The system checks the location provided by the Passenger and computes the corresponding zone in the city 7. The system, based on the city zone computed, retrieves the corresponding taxi queue 8. If there are no taxi in the queue: (a) Returns an error to the Passenger and invites him to retry later (b) Go back to Event flow 2 9. The system takes the first taxi in the queue 10. The system send the request to the taxi driver 11. If the taxi driver does not accept the request: (a) The system puts the Taxi driver at the bottom of the queue (b) Go back to Event flow 9 12. The system sends to the Taxi driver the location of the meeting point with the Passenger 13. The system computes an approximate waiting time for the Passenger 14. The system informs the Passenger of the forthcoming arrival of the Taxi and the approximate waiting time.
Exit condition	The passenger is waiting for the taxi driver and the taxi driver is reaching him

Use case

Request of a taxi (3)

Sequential Diagram



Use case

Reservation of a taxi (1)

Use case	**Reserve a taxi**
Actors	Passenger, Taxi Driver
Goals	A passenger must be able to reserve a taxi ride from an origin, to a destination at a specific time and date
Enter condition	• The Passenger must already be registered to the service • The Taxi driver must be registered to the service • The Passenger must already have opened the application (mobile or web) and logged in • The Taxi driver must be available
Event flow	1. The Passenger goes to the section for reserving a taxi ride 2. The Passenger fills the form with: origin, location, date, time, number of passengers 3. The Passenger submit the reservation to the system 4. If there are errors with the data (missing fields, not valid locations or number of passengers greater than 3) or the date and time provided are not 2 hours in advance: (a) The system notifies the error to the Passenger (b) Go back to Event flow 2

Use case

Reservation of a taxi (2)

5. The reservation is stored in the system
6. The system waits until 10 minutes before the date and time of the reservation
7. The system checks the origin location provided by the Passenger and computes the corresponding zone in the city
8. The system, based on the city zone computed, retrieves the corresponding taxi queue
9. If there are no taxi in the queue:
   (a) Wait until there is at least one taxi driver in the queue.
   (b) Go back to Event flow 9
10. The system takes the first taxi in the queue
11. The system send the request to the taxi driver
12. If the taxi driver does not accept the request:
    (a) The system puts the Taxi driver at the bottom of the queue
    (b) Go back to Event flow 9
13. The system sends to the Taxi driver the location of the meeting point with the Passenger
14. The system computes an approximate waiting time for the Passenger
15. The system informs the Passenger of the forthcoming arrival of the Taxi and the approximate waiting time.

Use case

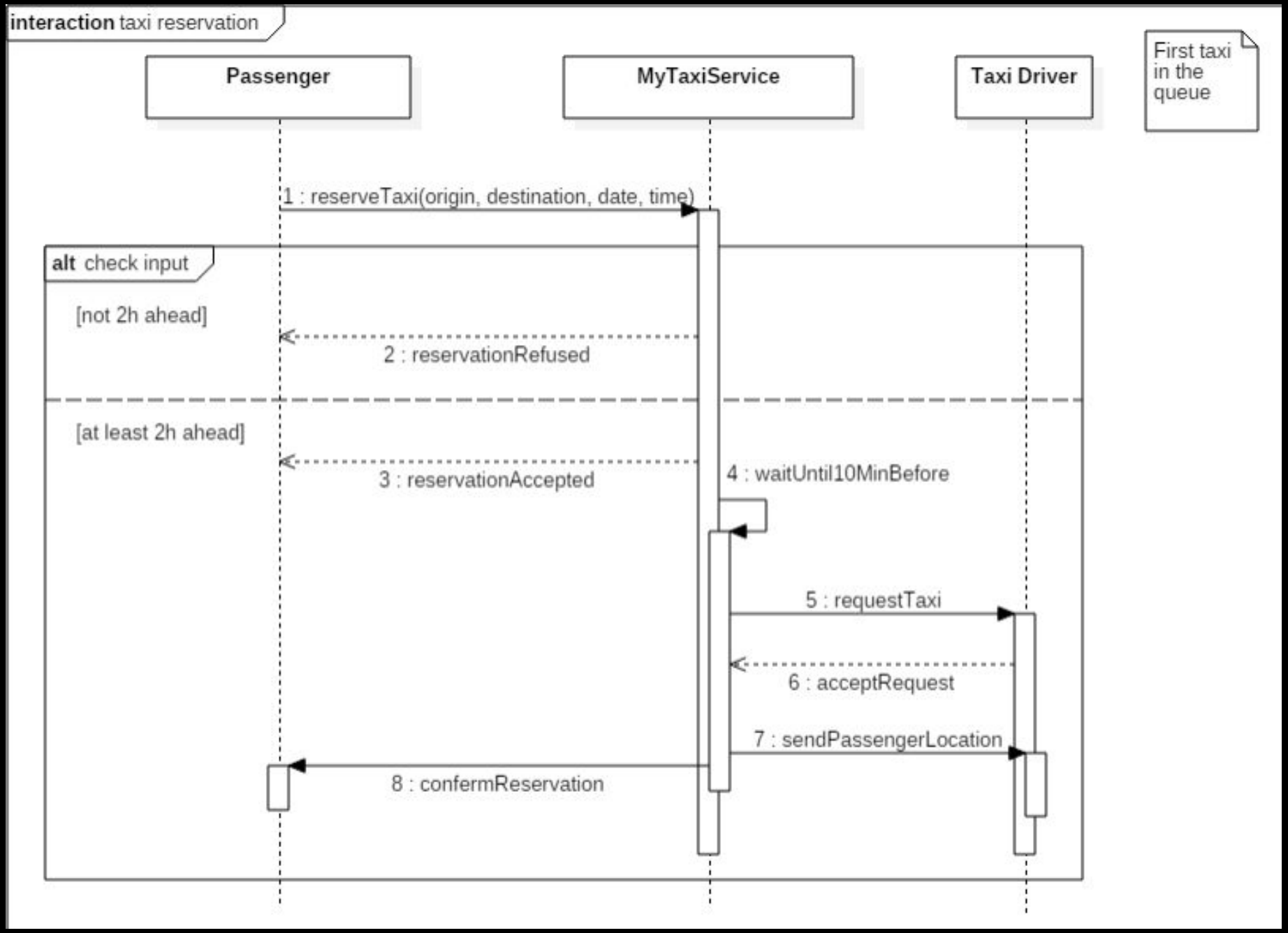
Reservation of a taxi (3)

Exceptions	The passenger can abort the procedure only during the waiting phase. If he decides to abort the procedure: 1. If the time of cancellation of the reservation is later than 10 minutes before the meeting time: (a) The system notifies the Passenger of the impossibility to cancel the reservation 2. The system cancel the reservation 3. The system notifies the Passenger of the successful deleting of the reservation
Exit condition	The passenger is waiting for the taxi driver and a Taxi driver is reaching him

Use case

Reservation of a taxi (4)

Sequential Diagram



Use case

Start working for a taxi driver

Use case	**Start working**
Actors	Taxi Driver
Goals	A Taxi Driver must be able to set himself as available (start working)
Enter condition	• The Taxi Driver must be already registered to the service • The Taxi Driver must already have opened the application
Event flow	1. The Taxi Driver press on the apposite button with the intention of setting himself as available 2. The system receives the updated status of the Taxi Driver 3. The system retrieves the location of the Taxi Driver from the GPS system 4. The system computes the actual zone of the Taxi Driver 5. The system puts the taxi driver at the bottom of the taxi queue relative to the computed taxi zone. 6. The system notifies the Taxi Driver and now he is waiting for a request
Exit condition	The Taxi Driver is waiting in a Taxi Queue

Use case

Accept request for a Taxi Driver (1)

Use case	**Accept a request**
Actors	Taxi Driver, Passenger
Goals	• A Taxi Driver must receive requests only if he is available • A Taxi Driver must be able to choose if he wants to accept an incoming request or not
Enter condition	The taxi driver must have already opened the application and must already be in a taxi queue

Use case

Accept request for a Taxi Driver (2)

Event flow	1. The system receives a request from a Passenger, computes the correspondent taxi zone and retrieves the taxi queue 2. The system picks the first taxi driver in the queue and forwards to the taxi driver the request 3. If the taxi driver refuses to accept the request: (a) The system puts the taxi driver at the bottom of the queue (b) Go back to Event flow 2 4. The system removes the taxi driver from the taxi queue and sets him as busy 5. The system communicates the taxi driver with information about the meeting point 6. The taxi driver reaches the meeting point and takes the passenger to the location he desires 7. The taxi driver set himself as available 8. Go to "Use case: Start Working", Event flow 2
Exit condition	A taxi driver is on his way to the meeting point with the passenger

Use case

Stop working for a taxi driver

Use case	**Stop working**
Actors	Taxi Driver
Goals	A Taxi Driver must be able to set himself as not available (stop working)
Enter condition	• The Taxi Driver must be already registered to the service • The Taxi Driver must already have opened the application • The Taxi Driver must be available
Event flow	1. The Taxi Driver press on the apposite button with the intention of setting himself as not available 2. The system receives the updated status of the Taxi Driver 3. The system removes the Taxi Driver from the queue 4. The system set the taxi driver as not available
Exit condition	The Taxi Driver is not available

Functional Requirements

- **Passenger**:
  - The system must not allow an already signed up Passenger to register himself (same username) again to the system
    - The username provided must be not empty
    - The password provided must be not empty
  - The system must notify the Passenger in case:
    - The username provided is empty
    - The password provided is empty
    - The username provided already exists in the system
  - The system must provide the Passenger with a way to abort the registration procedure
  - If a passenger makes a request and the corresponding taxi queue is not empty, then the system must sooner or later respond positively to the Passenger

Functional Requirements

- A passenger request must be refused **if and only if** there are no taxi driver available in the corresponding taxi queue or the number of passengers of the ride is greater than 3 (*)
- A reservation must be refused if and only if:
  - Origin and destination are the same location
  - The number of passengers of the ride is greater than 3
  - time(meeting time) - time(reservation) < 2 hours
- When the system looks for a taxi driver for serving a reservation (10 minutes before the meeting time), if the corresponding taxi queue is empty, it will wait until there is at least one taxi driver in the queue.

Functional Requirements

(*) We notice immediately that if all the taxi drivers in a queue refuse to accept the Passenger request, this will be forwarded again to the first taxi driver in the queue, who is the first one that refused it.

We think that this behavior is fair but can cause a loop and the Passenger could wait for a response a lot of time.

Functional Requirements

- Taxi Driver
  - Once available, he must be placed at the bottom of the queue of the zone corresponding to his location
  - Can be in one of this three states, which are mutually exclusive:
    - Available
    - Not Available
    - Busy
  - When “Available”, he must be exactly in one queue
  - When “Available”, he must receive requests only from passengers which specified a meeting point inside the correspondent taxi zone
  - When not “Available” (“Not Available” or “Busy”) must not be in any queue

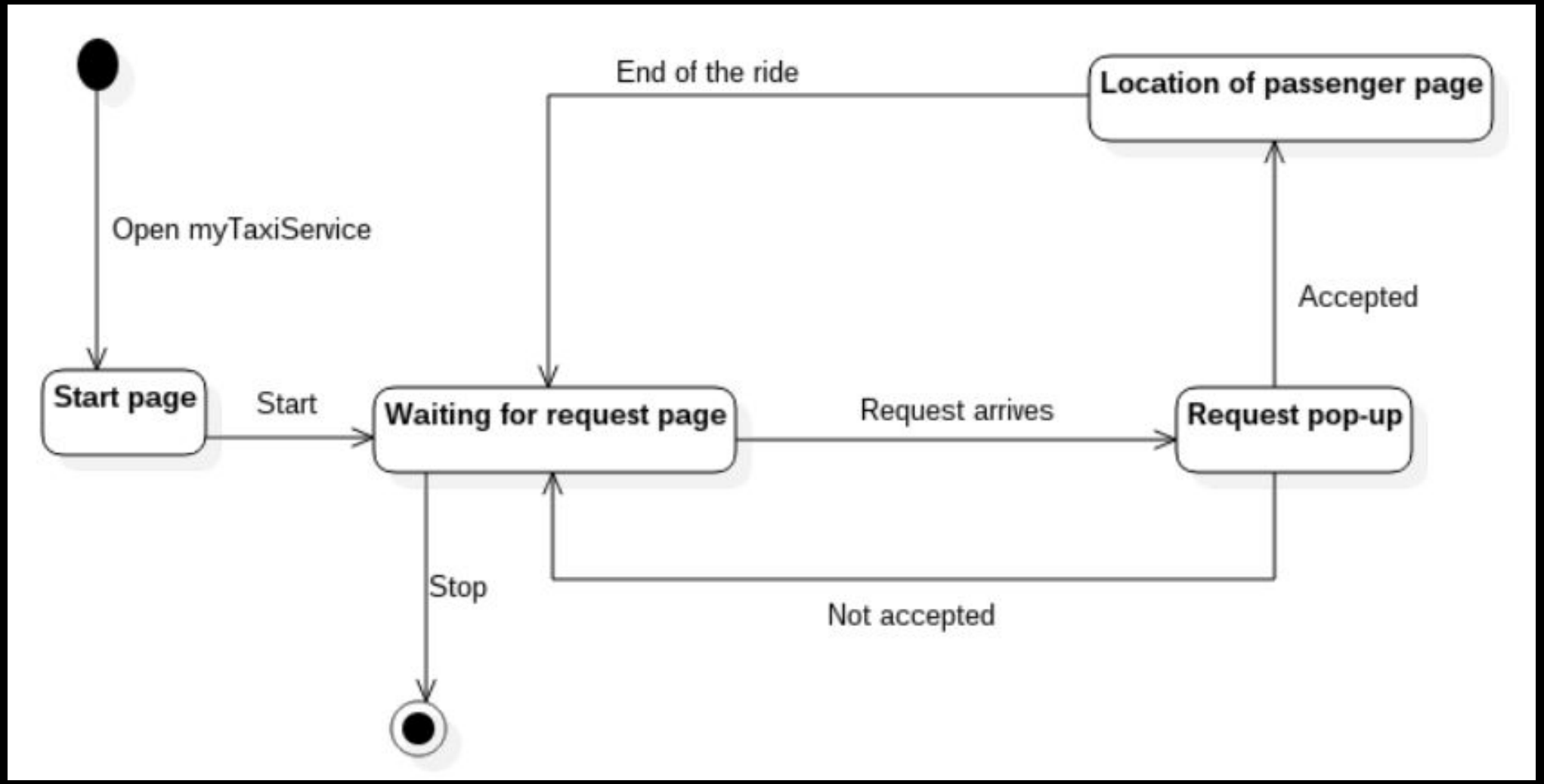
Functional Requirements

- Taxi Driver
  - Each queue can be empty or have a finite positive number of taxis
  - The system must put the taxi driver at the bottom of the queue if:
    - He refuses a Request
    - He does not respond to a request within the 10 seconds from the reception of it
  - Can receive requests only if he is at the top of the taxi queue
  - For each location of the taxi driver must correspond exactly one taxi zone
  - When a Taxi driver accepts a request from a Passenger, he must be removed from the corresponding taxi queue and set as busy

User interface

Taxi Driver

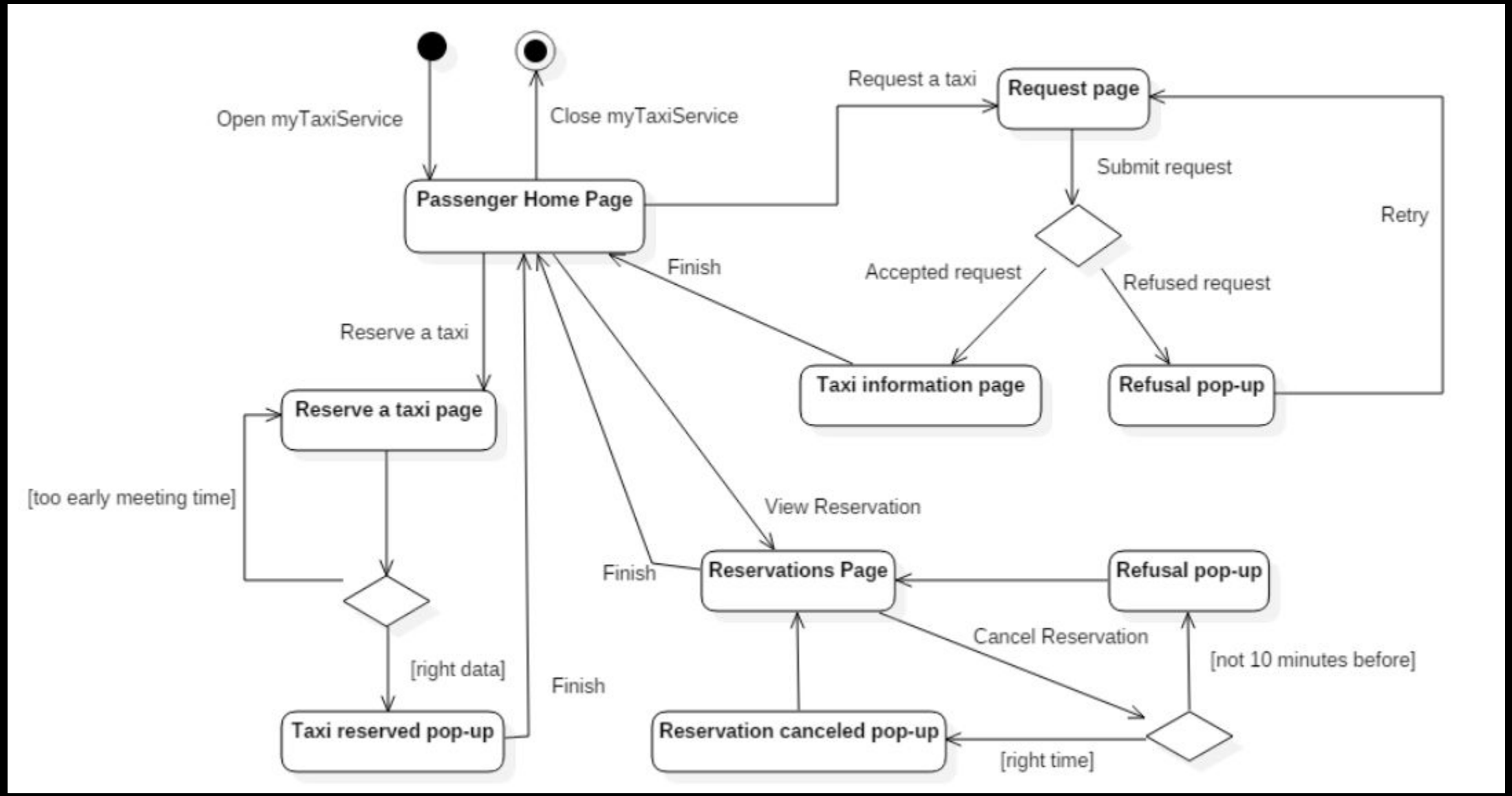
GUI statechart



User interface

Passenger

GUI statechart



Non Functional Requirements

- The performance of the web and mobile applications are strictly related on the performances of the internet connection of the devices they are running on. The interfaces must not cause any delay in the interaction.
- The system must work 24/24 7/7 with a maximum fault time of 1 hour / week

Appendix

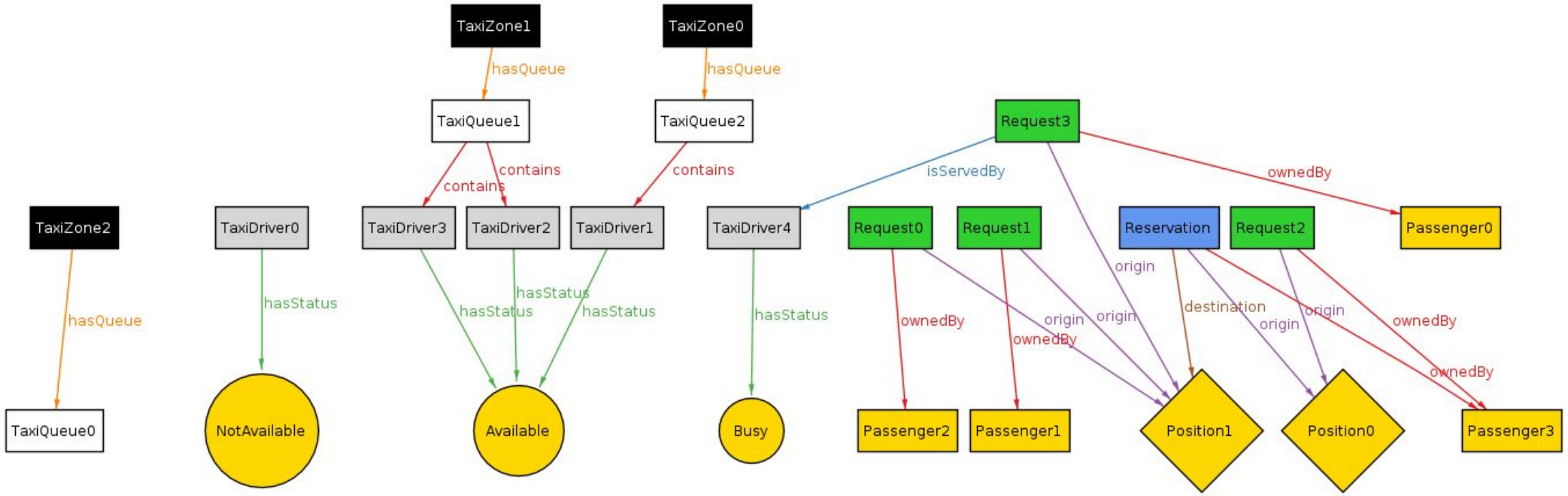
- Alloy
  - Prevalently used in the first part of the analysis
  - Strict conjunction with the diagram of entities
  - Modeled only the high-level requirements

Alloy

- ENTITIES:
  - Passenger
  - TaxiDriver
  - TaxiZone
  - TaxiQueue
  - Ride
    - Request
    - Reservation
  - (Position)
  - (Status)

- FACTS:
  - A passenger can have only one request at time
  - To one taxi zone corresponds exactly one taxi queue
  - A Taxi driver can be only in one queue
  - All the taxi driver that are in a queue are available and all the taxi that are available are in a taxi queue
  - A Taxi driver can serve a ride only if he is busy and a ride can be served only by busy taxi drivers
  - A Taxi driver can serve only one ride at time
  - The origin and the destination of a reservation must be different

Alloy


TaxiZone1
TaxiZone0
hasQueue
hasQueue
TaxiQueue1
TaxiQueue2
Request3
contains
contains
contains
isServedBy
ownedBy
TaxiZone2
TaxiDriver0
TaxiDriver3
TaxiDriver2
TaxiDriver1
TaxiDriver4
Request0
Request1
Reservation
Request2
Passenger0
origin
hasStatus
hasStatus
hasStatus
hasStatus
hasStatus
destination
origin
origin
origin
origin
ownedBy
ownedBy
ownedBy
ownedBy
hasQueue
TaxiQueue0
NotAvailable
Available
Busy
Passenger2
Passenger1
Position1
Position0
Passenger3

Thank you