Car Selling Application

Analysis and Design Document

Student: Abrudan Darius

**Group: 30233**

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 05/04/17 | 1.0 |  | Abrudan Darius |
| 25/05/17 | 2.0 |  | Abrudan Darius |
|  |  |  |  |
|  |  |  |  |

Table of Contents

I. Project Specification 4

II. Elaboration – Iteration 1.1 4

1. Domain Model 4

2. Architectural Design 5

2.1 Conceptual Architecture 5

2.2 Package Design 5

2.3 Component and Deployment Diagrams 6

III. Elaboration – Iteration 1.2 7

1. Design Model 7

1.1 Dynamic Behavior 7

1.2 Class Design 8

2. Data Model 8

3. Unit Testing 8

IV. Elaboration – Iteration 2 9

1. Architectural Design Refinement 9

2. Design Model Refinement 10

V. Construction and Transition 10

1. System Testing 10

2. Future improvements 11

VI. Bibliography 11

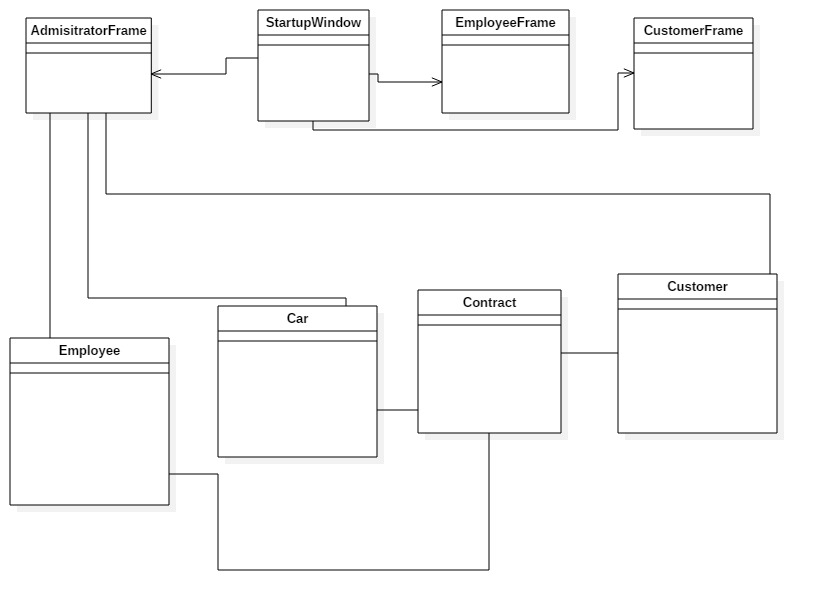
# Project Specification

Design and implement a client-server application for selling cars. The application will be used by customers, selling company’s employees and an administrator. A customer can search and view the available cars for a specific mark and model and also has the possibility of configure some additional parts for a car. He could filter the search results by price and year. In order to buy a car, a customer must fill a contract with its personal data, contract created by the employee. The administrator can make CRUD operations (create/update/delete) on cars, employees and customers.

# Elaboration – Iteration 1.1

# Domain Model

The domain model will look like the preliminary class diagram shown below:

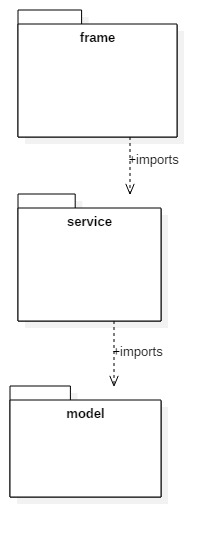


# Architectural Design

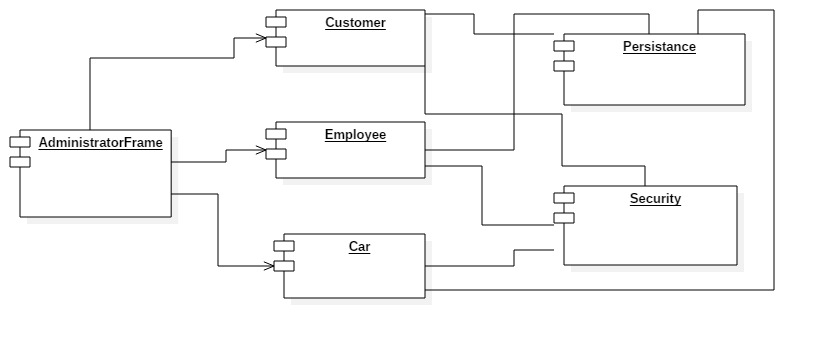
## Conceptual Architecture

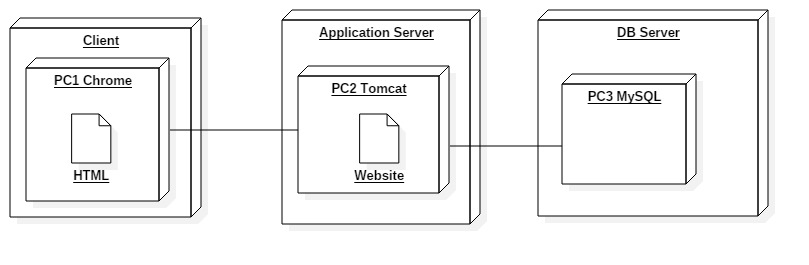
As an architectural pattern I choose to implement the client-server pattern for my car selling application with REST full web service based on JAX-RS specification. The server contains 5 services, User Services, Employee Services, Car Services, Contract Service and Customer Service, which contain the functionality of users, employees, cars, contracts and customers, services that can be accessed by the clients of the application, represented by the administrator, the employees and the customers.

## Package Design



## Component and Deployment Diagrams

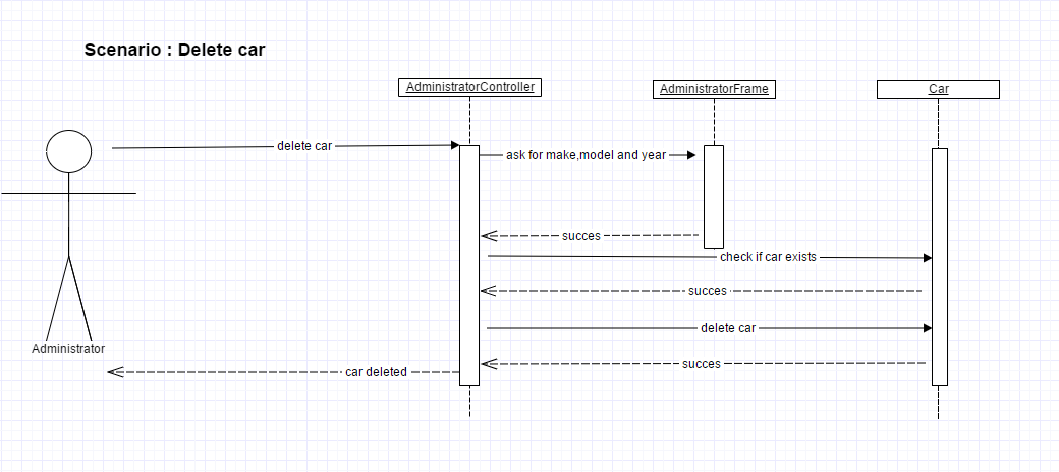


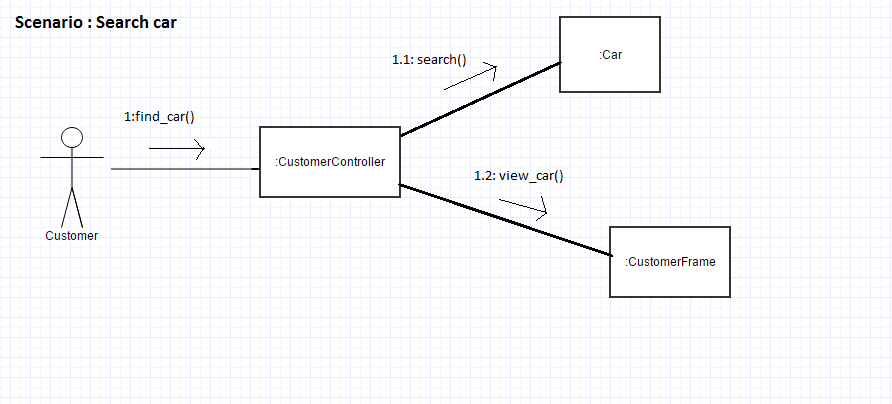


# Elaboration – Iteration 1.2

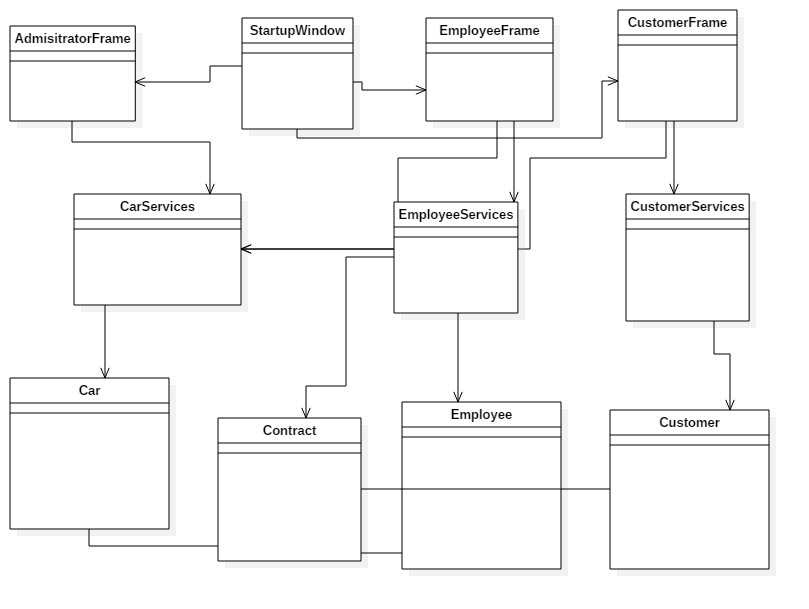
# Design Model

## Dynamic Behavio





## Class Design

**

# Data Model

The data model will consist of three tables in the DB: one for cars, one for users (employees and administrator) and one for the customers. The car table with contain the following attributes: make, model, year of fabrication, type of fuel, color, horsepower. The user table will have a username, a password and type, which will be either employee or admin. The customer table will have name, CNP and address as attributes.

# Unit Testing

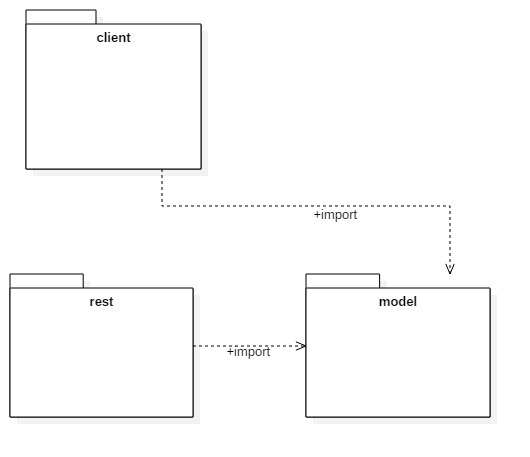
I designed a couple of tests such that all the inputs of the application (inputs introduced in J Text Fields) will be validated against invalid data before submitting the data and saving it in the database. If a Text Field is not completed, or if there is required a number and the user will introduce a string, it will output a J Option Pane with an appropriate message so that the user could know exactly which field was not completed correct. I also verified for the log in Startup Window such that the username exist and the password for a specific username is correct. In case of such an inconsistence, a message will appear on the screen.

# Elaboration – Iteration 2

# Architectural Design Refinement

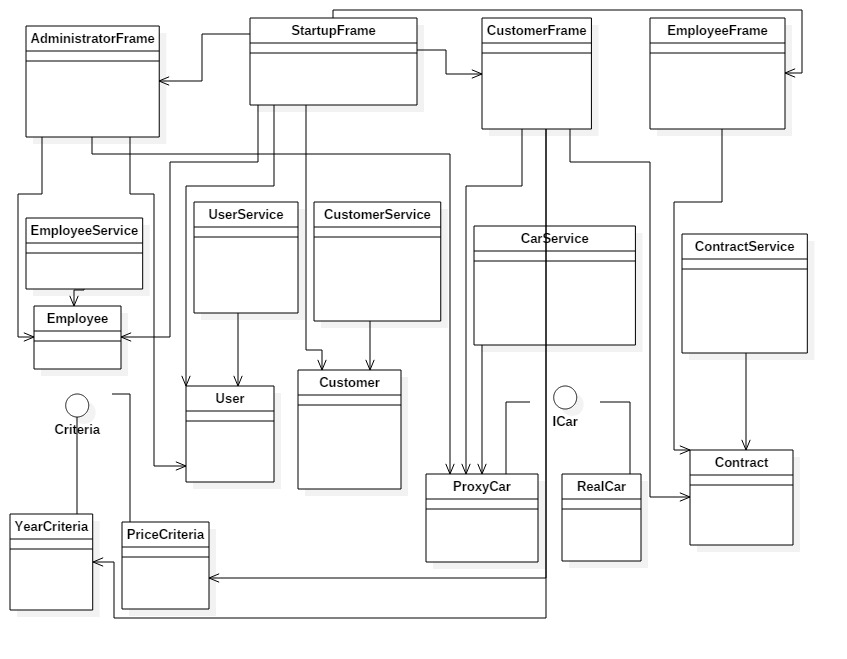
As a refinement to the architectural design, I choose to implement the Proxy design pattern for the car class, so I have I Car interface and 2 classes, Real Car and Proxy Car, the last one is a proxy class to reduce memory footprint of Real Car object image field loading. Another design pattern that is present in my architecture is Filter design pattern or so called Criteria DP. I use the Filter DP to filter the results of search when a customer search for a car, so he has the possibility of choosing the maximum price that he will pay for a car, or the minimum year required for the car fabrication in order to buy it.

The component and deployment diagrams remain the same as they were in previous steps of implementation, but there is a modification regarding the package diagram. The new one is shown below:



# Design Model Refinement

The final class diagram after the process of implementation is the following:

**

# Construction and Transition

# System Testing

I test the functionality of the car selling application created and it works as I planned. I test the scenarios for filter the process of search for the customers with different prices and years. I test also the possibility of searching by make and model, when a car is found with the specific data, a picture of the car can also be seen. For all the methods implemented which need some data from the graphic interface (inputs introduced in J Text Fields), I designed some logic so they will be validated against invalid data before submitting the data and saving it in the database.

# Future improvements

As some future improvements of the project, I think that the database should be extended with a lot more make and models of cars, and also with more specifications for the cars. Also the application should permit the customer to search by multiple criteria, not just year and price, which are the most relevant in my opinion. Another improvement could be to have web clients instead of application clients.

# Bibliography

1. <http://www.mkyong.com/tutorials/jax-rs-tutorials/>
2. <https://stackoverflow.com/>
3. <https://docs.oracle.com/javase/tutorial/>
4. <https://www.tutorialspoint.com/>