Analysis Document:
Car Rental Service

Group 4
CSCI 4050: Software Engineering
Instructor: Krys Kochut
Osama Mansour, Stephen Patton, Vincent Lee, and Minh Pham

Table of Contents

1. Introduction....1
1.1 Purpose of the system....1
1.2 Scope of the System....1
1.3 Objectives and Success Criteria....1
1.4 References....2
1.5 Overview....2
2. Proposed System....2
2.1 Overview....2
2.2 Functional Requirements....2
2.3 Non-Functional Requirements....3
2.3.1 Performance....3
2.3.2 Reliability....3
2.3.3 Maintainability....3
2.3.4 Implementation....3
2.3.5 Extensibility & Expandability....3
2.4 System Model....4
2.4.1 Scenarios....4
2.4.2 Use Case Model....21
2.4.2.1 ReturnVehicle....22
2.4.2.2 BrowseVehicles....23
2.4.2.3 RegisterCustomer....24
2.4.2.4 AddVehicle....25
2.4.2.5 ListVehicle....27
2.4.2.6 ModifyVehicle....28
2.4.2.7 DeleteVehicle....29
2.4.2.8 AddVehicleType....30
2.4.2.9 ListVehicleType....32
2.4.2.10 ModifyVehicleType....33
2.4.2.11 DeleteVehicleType....34
2.4.2.12 ListProfile....35
2.4.2.13 ModifyProfile....36
2.4.2.14 DeleteProfile....37
2.4.2.15 Login....38
2.4.2.16 LogOut....39
2.4.2.17 AddUser....40
2.4.2.18 ListUser....41
2.4.2.19 ModifyUser....42
2.4.2.20 RemoveEmployee....43
2.4.2.21 TerminateMembership....44
2.4.2.22 CancelReservation....45
2.4.2.23 ReserveCar....47
2.4.3 Object Model....48

2.4.3.1 Data Dictionary 48
2.4.3.1.1 Entity Package 48
2.4.3.1.2 user.control package 57
2.4.3.1.3 user.boundary package 58
2.4.3.1.4 administrator.control package 60
2.4.3.1.5 administrator.boundary package 62
2.4.3.1.6 employee.control package 64
2.4.3.1.7 employee.boundary package 66
2.4.3.1.8 customer.control package 67
2.4.3.1.9 customer.boundary package 68
2.4.3.2 Class Diagrams 70
2.4.3.2.1 Package Diagram 70
2.4.3.2.2 Entity Package 71
2.4.3.2.3 user.control package 72
2.4.3.2.4 user.boundary package 72
2.4.3.2.5 administrator.control package 72
2.4.3.2.6 administrator.boundary package 73
2.4.3.2.7 employee.control package 73
2.4.3.2.8 employee.boundary package 73
2.4.3.2.9 customer.control package 74
2.4.3.2.10 customer.boundary package 74
2.4.4 Dynamic Model – Sequence & Communication Diagrams 75
2.4.4.1 ReturnVehicle – Use Case: Page 22 75
2.4.4.2 RegisterCustomer – Use Case: Page 24 77
2.4.4.3 AddVehicle – Use Case: Page 25 79
2.4.4.4 ListVehicle – Use Case: Page 27 81
2.4.4.5 ModifyVehicle – Use Case: Page 28 83
2.4.4.6 DeleteVehicle – Use Case: Page 29 85
2.4.4.7 AddVehicleType – Use Case: Page 30 87
2.4.4.8 ListVehicleType – Use Case: Page 32 89
2.4.4.9 ModifyVehicleType – Use Case: Page 33 91
2.4.4.10 DeleteVehicleType – Use Case: Page 34 93
2.4.4.11 ListProfile – Use Case: Page 35 95
2.4.4.12 ModifyProfile – Use Case: Page 36 97
2.4.4.13 DeleteProfile – Use Case: Page 37 99
2.4.4.14 Login – Use Case: Page 38 101
2.4.4.15 LogOut – Use Case: Page 39 103
2.4.4.16 AddUser – Use Case: Page 40 105
2.4.4.17 ListUser – Use Case: Page 41 107
2.4.4.18 ModifyUser – Use Case: Page 42 109
2.4.4.19 RemoveEmployee – Use Case: Page 43 111
2.4.4.20 TerminateMembership – Use Case: Page 44 113
2.4.4.21 CancelReservation – Use Case: Page 45 115

2.4.4.22 ReserveCar – Use Case: Page 47 117
2.4.5 Navigation path and User Interface 119
2.4.5.1 Navigation path 119
2.4.5.2 User Interfaces Mock-Up 120
2.4.5.3 Registration of User Profile 120
2.4.5.4 Registration of User Address Information 120
2.4.5.5 Choose of Driving Plans 121
2.4.5.6 Registration of Payment Information 121
2.4.5.7 Sign Up Confirmation 122
2.4.5.8 AddUser 122
2.4.5.9 AddVehicles 123
2.4.5.10 Search Vehicle By Location 123
2.4.5.11 Admin Membership Search 124
2.4.5.12 CustomerView 124
2.4.5.13 Vehicle Search 125
2.4.5.14 Car Reservation 126
2.4.5.15 Car Reservation Information 126
2.4.5.16 Reservation Confirmation 127
3. Glossary ....

1. Introduction

1.1 Purpose of the system

The purpose of the project is to develop a good online car rental service for young drivers. The car company uses very flexible rental agreements so that both people under age 25 and drivers older than 25 can rent. The company will be only available online. The system will have employee and administrator users with different levels of access. Although administrators and employees can both check the database, the administrators have more access and check on the employees. Customers can sign up, log in, search a car, reserve a car, pay rental fees, and terminate their accounts. The system allows multiple users to access and search for cars concurrently.

1.2 Scope of the System

The proposed system will be able to deal with all aspect of car rental and management. Users can check which car is available and at which location. Drivers can request a vehicle for a certain date and time to pick up at a specific location. If the requested car is not available at a desired location, the system can also check whether a similar car is available at an alternate location. The key stakeholders of the system are: customers, employees and administrators, and their interest to the system are listed as below:

Stakeholders	Interests
Customers	Register his or her membership Search for a car and check the availability Reserve a car, pay the fee, and cancel the reservation Having a stable system, e.g. what if the system crashes while taking a reservation
Employees	Manage a car rental, e.g. view which car is available at which location Manage a customer: email to remind the driver to pay a fee
Administrators	Manage the whole system, e.g. add more administrators, hire and add an employee, add more cars or another location

1.3 Objectives and Success Criteria

- The system should successfully allow an administrator to add, modify, delete, and search/ list entries of other administrators, rental locations, cars, employees, and customers. They should be able to define and maintain a number of vehicle types, enter individual vehicles and specify their types, prices and other important properties. They can also manage the employees system, add and modify an employee. They should be able to remove a customer for serious violations of the rental agreement.
- The system should successfully allow an employee to view the information and car rental history of all customers; he can also view all the car, availability and locations, and email

to notify customers about reservation, late fee, cancellation, and answer some specific questions.

- The system should successfully allow a customer to create an account and pay the initial 6-month membership fee. It also allows customers to view all available cars at all locations, and to register for a car as well as cancel the reservation. If the requested vehicle is not available, the system can suggest a similar vehicle at a different location. The reservation must be specific in vehicle type, pickup time and the length of the rental. The customer can also provide information about the condition of the returned vehicle and give comments about the vehicle and the rental service if desired. Finally, they can renew the membership as well as terminate their account**.**

1.4 References

- The project will be in Java or C++ specification
- Persistent data store with MySQL
- The system use accepted standards (HTML,SQL, …)
- The system must be accessible from a common Web browser (such as Mozilla Firefox, Google Chrome and Microsoft Internet Explorer)

1.5 Overview

The system being developed will be an easy to use online reservation system which supports all the requirements described in this document. It also will be capable to maintain data integrity while allowing a large number of drivers to access concurrently.

2. Proposed System

2.1 Overview

The proposed system uses flexible rental agreements and allows young and old drivers to rent a car as long as they can confirm their driving credentials. It will be available online. It will allow drivers to search, register a vehicle, cancel a reservation, and pay fees. Administrators will be allowed to manage the whole system, such as information regarding drivers, vehicles, locations, employees, and to ensure the stability of the system. Also, the system can be accessed by a large amount of users (administrators, employees and drivers) at the same time. The system provides an easy-to-use interface that is compatible to most of web browsers.

2.2 Functional Requirements

- Administrator should be able to add, remove, and edit all vehicle information, customer information, employee information, and business pricing information as needed.
- Employee should be able to add, remove, and edit all customer information on the condition that they have the customer's name and driver’s license information.

Employees should also be able to add, remove, and edit particular instances of vehicles but not generic types.

- Customers should be able to search for and reserve available cars and should be able to edit their personal information, credit card information, and rental history. Customers should be able to cancel reservations within a certain time period, access billing information for past and present rentals, return a vehicle, and terminate membership through the car rental service system at any time.

2.3 Non-Functional Requirements

2.3.1 Performance

- The system should be provide the ability to modify, delete, search, and add users, vehicles, and locations in-line with real world transactions.
- Allow for multi-user access with no noticeable delay.

2.3.2 Reliability

- The system should work 24hrs a day, seven days a week.
- Maintain as close to 100% uptime SLA (service level agreement). Maintenance included, the system should have near 99.99% uptime
- The system will allow employees access to customer accounts when they provide a name and driver's license

2.3.3 Maintainability

- The system should allow for system administrators should be able to make changes to any of the information currently stored in the system.
- The system must use a persistent data store for all the relevant data.
- Databases will make nightly backups of all data.

2.3.4 Implementation

- The system back-end should be created using Java or C++ with a MySQL database.
- The system front-end should be created using accepted standards including HTML, CSS, Jquery, etc.
- The system should work on the most popular current major browsers.

2.3.5 Extensibility & Expandability

- The system should be developed in a way to accommodate future functionality and expansion into variety of platforms.

2.4 System Model

2.4.1 Scenarios

Scenario name	administratorAddVehicleType
Participating actor Instances	John: Administrator
Flow of events	1. John would like to add a new vehicle type to system. John logs into the system and selects the vehicles page. 2. John selects the add vehicle type button and fills in the required fields in the provided form: name, price, etc. John clicks continue. 3. The system displays the new vehicle type with provided information. Once John is satisfied with this new vehicle type he clicks save to create the new vehicle type.

Scenario name	administratorRemoveVehicleType
Participating actor Instances	John: Administrator
Flow of events	1. John would like to remove a poorly performing vehicle type. John logs into the systems and selects the vehicles page. 2. John selects the vehicle type to be removed in a drop down box and clicks remove. 3. The system displays a page listing the vehicle type to be removed and the current vehicles that will be no longer available. Once John is satisfied with these changes, he clicks save and the changes become effective.

Scenario name	administratorSetRentalPrice
Participating actor Instances	John: Administrator
Flow of events	1. John wants to set the rental price of a vehicle type. Once logged into the system, he navigates to the pricing page and selects vehicles. 2. John chooses the vehicle type and is presented with a form containing the current price of hourly and daily rentals. John enters in new rates for one or both of the fields. John clicks continue. 3. The system displays a confirmation screen with the current vehicle pricing and the new pricing John set. John reviews these changes and clicks save to initiate the changes.

Scenario name	**administratorSetMembershipPrice**
Participating actor Instances	John: Administrator
Flow of events	1. John would like to change the price of the membership tiers. After logging in, he navigates to the pricing page and selects memberships. 2. John chooses the tier he would like to update pricing on, and is presented with a form containing the current price. John enters in a new price and clicks continue. 3. The system displays a confirmation screen with the current tier's price and the new pricing John has set. John reviews these changes and clicks save to activate the changes.

Scenario name	**administratorAddRentalLocation**
Participating actor Instances	John: Administrator
Flow of events	1. John is expanding his business and would like to add a new rental location. Once logged in, he selects the locations page. 2. John then chooses add rental location. He is presented with a form containing: name, address, vehicle capacity, etc. John fills in the required information and clicks continue. 3. The system displays a confirmation screen with the new locations information. John reviews the information. Once he is satisfied, he clicks save to create the new rental location.

Scenario name	**administratorModifyRentalLocation**
Participating actor Instances	John: Administrator
Flow of events	1. John needs to modify a rental location's information because his landlord's new contract cuts his lot substantially. John logs into the system and visits the rental location page. 2. John selects the rental location through a dropdown box and clicks edit. He is presented with the location's property information in a form. He modifies the appropriate values, and clicks continue. 3. The system displays a confirmation screen showing location's old properties and the new ones. Once John is satisfied with these changes he clicks save to make them effective.

Scenario name	**administratorRemoveRentalLocation**
Participating actor Instances	John: Administrator
Flow of events	1. John is notified by a landlord that the contract for a particular rental location will not be renewed. John logs into the system and navigates to the rental location page. 2. John selects the rental location to be removed from a drop down list, and clicks remove. 3. The system displays a confirmation screen showing the location to be removed. Once John has verified that the location to be removed is correct, he clicks save to remove the rental location.

Scenario name	**administratorAddVehicle**
Participating actor Instances	John: Administrator
Flow of events	1. John recently purchased additional vehicles and they have been prepared at the respected rental locations. John would like to add them to the system, and allow customers to begin renting them out. 2. John logs in and navigates to the vehicles page. John clicks the add vehicle button. He is presented with a form containing: vehicle type, make, model, year, registration tag, current mileage, date last serviced, condition, etc. John fills out this required information and clicks continue. 3. The system displays a confirmation screen showing the new vehicle and its information. John reviews this information. Once verified, he clicks save to add the new vehicle to the fleet.

Scenario name	**administratorModifyVehicle**
Participating actor Instances	John: Administrator
Flow of events	1. John just got a vehicle serviced, and needs to add this information to the vehicle records. He logs into the system, and selects the vehicles page. 2. John searches for the vehicle through a search bar, or locates it through its current location. John clicks modify on the vehicle profile, and is presented with a forum containing the vehicle information. He updates the last serviced date, and clicks continue. 3. The system displays a confirmation screen showing the changes made to the vehicle. John reviews these changes, and once verified, he clicks save to update the vehicle's information.

Scenario name	**administratorRemoveVehicle**
Participating actor Instances	John: Administrator
Flow of events	1. John finished a performance review, and found a vehicle that is unprofitable and would like to remove it from the system. He logs in and navigates to the vehicles page. 2. John searches for the vehicle using a search bar and clicks modify on the vehicle profile. He is then presented with a form with the vehicle's information. At the bottom there is a button to remove the vehicle. He clicks the remove button. 3. The system displays a confirmation screen showing the vehicle to be removed. John reviews these changes, and clicks save to remove the vehicle from the fleet.

Scenario name	**administratorTerminateUser**
Participating actor Instances	John: Administrator
Flow of events	1. John finds out that a customer damaged a vehicle and failed to report the damage. 2. John logs into the site and navigates to customers and searches for the customer name. Once a match is located he clicks modify. He is brought to a form containing the users profile information. At the bottom he clicks disable user. 3. The system displays a confirmation screen showing the user and information that is going to be disabled. Once this information is verified, John clicks save, and the user can no longer use the service.

Scenario name	**administratorModifyCustomerInformation**
Participating actor Instances	John: Administrator
Flow of events	1. John needs to help a customer modify some information on their account. The customer says the system is not working. John logs into the system and navigates to the customer page. 2. John locates the customer profile and verifies the customer through information on file. John clicks on the edit tab on the profile and is directed to a form with the customer's information. He modifies the requested fields, and clicks continue. 3. The system displays a confirmation screen with the previous information and information updated. Once John verifies the changes are correct, he clicks save to employ the changes.

Scenario name	**administratorAddEmployee**
Participating actor Instances	John: Administrator
Flow of events	1. John is hiring new employees to support his company. He logs into the system, and navigates to the employee page. 2. John clicks add employee button and is presented with a form. John types in the employee information: name, username, and temporary password. John clicks the continue button. 3. The system displays a confirmation screen of the new employee to be added. John checks for errors and once satisfied, clicks the create button to add the new employee.

Scenario name	**administratorModifyEmployeeInformation**
Participating actor Instances	John: Administrator
Flow of events	1. John needs to modify the information of an employee in the system. He logs into system, and navigates to the employee page. He then performs a search for the employee. 2. John locates the employee's profile. He clicks modify and is directed to a page with the employee's information displayed in a form. He makes the necessary changes and clicks continue. 3. The system displays a confirmation screen with the previous record, and the updated version. John verifies that there are no errors, he clicks save to update the employee's information.

Scenario name	**administratorAddAdministrator**
Participating actor Instances	John: Administrator
Flow of events	1. John would like to create an administrator account for his business partner. He logs into the system, and navigates to the administrator page. 2. John clicks the add additional administrator. He is presented with a form with fields: name, username, temporary password, etc. John fills out the required information and clicks the continue button. 3. The system displays a confirmation screen showing the administrator to be added to the system. Once John verifies the information, he clicks create to add the new administrator.

Scenario name	**administratorRemoveEmployees**
Participating actor Instances	John: Administrator
Flow of events	1. John would like to remove an employee who quit the company. He logs into the system, and navigates to the employee page. 2. John searches for the employee in question. Once the employee is found, he clicks modify and at the bottom of the page, is a delete button. John clicks the delete button. 3. The system displays a confirmation screen showing the user and profile information to be deleted. Once John verifies that the user is correct, John clicks delete, and the employee is removed from the system.

Scenario name	**customerLogIn**
Participating actor Instances	Paul: Customer
Flow of events	1. Paul opens the web page and types in his username and password. 2. The system loads the customer home page and displays it to Paul.

Scenario name	**customerLogOut**
Participating actor Instances	Paul: Customer
Flow of events	1. Paul is finished using the website, locates the logout button at the top of the page, and clicks on it. 2. The system returns to the login screen.

Scenario name	**customerRegisterNewID**
Participating actor Instances	Paul: Customer
Flow of events	1. Paul opens the webpage and clicks on the register new user link 2. The system loads the registration page which will prompt Paul for a username, password, full name, email, home address, phone number, driver's license number, and credit card number. It will also ask Paul to pay the membership fee. 3. Paul will enter all of the fields, agree to pay the fee, and click the register button 4. The system will add Paul to its database

Scenario name	**customerRenewMembership**
Participating actor Instances	Paul: Customer
Flow of events	1. Paul clicks on the “Renew/Extend Membership Fee” link on the 'My Account' page 2. Paul selects how long he wants to extend his membership. 3. The system asks Paul to confirm his purchase and Paul clicks on confirm 4. The system updates Paul's information and prints a confirmation message

Scenario name	**customerTerminateMembership**
Participating actor Instances	Paul: Customer
Flow of events	1. Paul clicks on the 'Cancel Membership' link on the 'My Account' page 2. The system displays the following message, “Once you cancel your membership, it cannot be undone and no money will be refunded. If you plan on using this service in the future you will have to make a new account. Are you sure you want to cancel your membership?” and prompts the user with a yes and no option 3. Paul clicks on 'yes'. 4. The system deletes Paul from the database

Scenario name	**customerModifyName**
Participating actor Instances	Paul: Customer
Flow of events	1. Paul clicks on the 'Update Information' link on the 'My Account' page 2. The system prompts Paul with the following prompt: “Which do you wish to modify?” and lists several options including 'Name' 3. Paul clicks on the 'Name' option 4. The system displays his current name as it is in the system along with a cancel button if Paul changes his mind. It also provides a space for Paul to enter his new name and a space for him to confirm his name 5. Paul enters his name and clicks the 'Update Name' link 6. The system updates Paul's name and displays a confirmation message

Scenario name	**customerModifyPhoneNumber**
Participating actor Instances	Paul: Customer
Flow of events	1. Paul clicks on the 'Update Information' link on the 'My Account' page 2. The system prompts Paul with the following prompt: “Which do you wish to modify?” and lists several options including 'Phone Number' 3. Paul clicks on the 'Phone Number' option 4. The system displays his current phone number as it is in the system along with a cancel button if Paul changes his mind. It also provides a space for Paul to enter his new phone number and a space for him to confirm his phone number 5. Paul enters his name and clicks the 'Update Phone Number' link 6. The system updates Paul's phone number and displays a confirmation message

Scenario name	**customerModifyAddress**
Participating actor Instances	Paul: Customer
Flow of events	1. Paul clicks on the 'Update Information' link on the 'My Account' page 2. The system prompts Paul with the following prompt: “Which do you wish to modify?” and lists several options including 'Address' 3. Paul clicks on the 'Address' option 4. The system displays his current address as it is in the system along with a cancel button if Paul changes his mind. It also provides a space for Paul to enter his new address 5. Paul enters his name and clicks the 'Update Address' link 6. The system updates Paul's address and displays a confirmation message

Scenario name	**customerModifyDriversLicense**
Participating actor Instances	Paul: Customer
Flow of events	1. Paul clicks on the 'Update Information' link on the 'My Account' page 2. The system prompts Paul with the following prompt: “Which do you wish to modify?” and lists several options including 'Driver's License Information' 3. Paul clicks on the 'Driver's License Information' option 4. The system displays his current driver's license information as it is in the system along with a cancel button if Paul changes his mind. It also provides a space for Paul to enter his new driver's license information. 5. Paul enters his Information and clicks the 'Update Driver's License Number' link 6. The system updates Paul's driver's license number and displays a confirmation message

Scenario name	**customerModifyEmail**
Participating actor Instances	Paul: Customer
Flow of events	1. Paul clicks on the 'Update Information' link on the 'My Account' page 2. The system prompts Paul with the following prompt: “Which do you wish to modify?” and lists several options including 'Email' 3. Paul clicks on the 'Email' option 4. The system prompts Paul to type in his current email, his new email, and a confirmation of his new email 5. Paul enters his information and clicks the 'Update Email link 6. The system updates Paul's email and displays a confirmation message

Scenario name	**customerModifyCreditCard**
Participating actor Instances	Paul: Customer
Flow of events	1. Paul clicks on the 'Update Information' link on the 'My Account' page 2. The system prompts Paul with the following prompt: “Which do you wish to modify?” and lists several options including 'Credit Card' 3. Paul clicks on the 'Credit Card' option 4. The system asks for Paul's new credit card information including: 16 digit number, expiration date, full name, billing address, and security code 5. Paul enters the information and clicks on 'Submit' 6. The system updates Paul's information and displays a confirmation message

Scenario name	**customerModifyPassword**
Participating actor Instances	Paul: Customer
Flow of events	1. Paul clicks on the 'Update Information' link on the 'My Account' page 2. The system prompts Paul with the following prompt: “Which do you wish to modify?” and lists several options including 'Password' 3. Paul clicks on the 'Password' option 4. The system prompts Paul to type in his current password, his new password, and a confirmation of his new password 5. Paul enters his information and clicks the 'Update Password' link 6. The system updates Paul's password and displays a confirmation message

Scenario name	**customerReserveVehicle**
Participating actor Instances	Paul: Customer
Flow of events	1. Paul is on the page of a vehicle he wants to reserve and clicks on the 'Reserve Vehicle' link 2. The system loads a page that prompts Paul for the date, time, duration, and drop-off destination for the rental of the vehicle. 3. Paul fills in the required information and clicks submit. 4. The system displays the following information: • Rental site • Car information • Date and time of pick up and drop off • Drop-off destination • Price The system asks Paul if this information is correct and to confirm payment 5. Paul confirms the information by clicking 'Confirm' 6. The system updates that Paul has made this reservation and displays a message confirming the reservation

Scenario name	**customerCancelReservation**
Participating actor Instances	Paul: Customer
Flow of events	1. Paul clicks on the 'Cancel Reservation' link in the “My Account” page 2. The system displays all the reservations currently on Paul's account and asks him which one he would like to cancel 3. Paul selects the reservation he wishes to cancel 4. The system asks Paul if he is sure and if it is less than an hour before the reservation time reminds Paul that there will be a fee charged to his account for late cancellation. 5. Paul clicks yes that he is sure 6. The system updates the information and displays a confirmation message

Scenario name	**customerNotifyVehicleReturn**
Participating actor Instances	Paul: Customer
Flow of events	1. Paul clicks on the 'Notify Vehicle Return' Link on the 'My Account' page 2. The system displays the Vehicle information and return site and prompts Paul to verify that it is correct 3. Paul clicks on yes 4. The system updates the return and displays a confirmation message

Scenario name	**customerBrowseVehicle**
Participating actor Instances	Paul: Customer
Flow of events	1. At the home screen Paul clicks on browse vehicles 2. The system provides the following optional constraints to Paul: • Type of vehicle • Location • Date and time of pick up and drop off • Price range 3. Paul chooses the constraints that he wants and clicks browse 4. If there are cars available that match Paul's constraints, the system displays them. It will display the car information, location, dates and times available, and price

Scenario name	customerViewRentalHistory
Participating actor Instances	Paul: Customer
Flow of events	1. From the 'My Account' page, Paul clicks on 'View Rental History'. 2. The system displays all previous rentals and any current reservations

Scenario name	employeeLogIn
Participating actor Instances	Mike: Employee
Flow of events	1. Mike enter the website address and opens the web page. Mike types in his user name and password. 2. The System presents mike with an employee view of the Car Rental Service system to perform his tasks

Scenario name	employeeLogOut
Participating actor Instances	Mike: Employee
Flow of events	1. Mike finishes his duties for the day and wants to logout of the Car Rental Service system. Mike looks to the top of the screen display, and finds and clicks the logout button. 2. The System will display a success message that mike has logged out of the system.

Scenario name	employeeAddVehicle
Participating actor Instances	Mike: Employee
Flow of events	1. Mike logs into the Car rental service system and wants to add a new vehicle. Mike locates the vehicle section and click the option of add a new vehicle. 2. The System then displays a new submission sheet requesting all required vehicle information including the vehicle name, type, make, model, year, and vin number. It also displays additional notes about the system. Mike fills out all the required fields and clicks the submit button. 3. The System updates the database and shows a success message displaying the new vehicle information. 4. Mike will then click the add another vehicle option to continue adding vehicles or will exit the vehicle success message and go back to the employee home view screen.

Scenario name	**employeeModifyCustomerInformation**
Participating actor Instances	Mike: Employee
Flow of events	1. Mike Logs into the car rental service system and wants to modify customer information at the request of a customer. He searches the customer in the database, and choose the edit customer information button. 2. The System will then prompt mike for a name and license number supplied by the customer. Mike types in the information and presses submit. 3. The System will display the existing selected customer information including customer name, address, credit card information, phone number, activity status, license, and login information. Mike will selected the instructed fields to edit, and will click a Save button 4. Bob will then continue by pressing a Submit to finalize the current modification or cancel to undo the modifications.
Scenario name	**employeeSearchCustomerInformation**
Participating actor Instances	Mike: Employee
Flow of events	1. Mike's Logged into the System, and he wants to search a customer's information. Mike will click on the Search for customers button on the main page of the system 2. The system will direct him to a display screen where he will be given a search bar. Mike then will type in the first and last name of the desired customer. 3. The system will then locate and display the customer with all of his/her information and vehicle rental history.

Scenario name	**cancelReservation**
Participating actor Instances	Mike: Employee
Flow of events	1. Mike logs into the System, and he wants to cancel a reservation on the request of a customer. Mike will search for a customer by clicking a button on the main page. 2. The system will direct him to a display screen where he will be given a search bar. Mike will type in the first and last name of desired customer for cancellation. 3. The system will locate and display the customer with all of his/her information along with vehicle rental history. Mike will then find the requested vehicle to cancel. 4. Mike will then click the cancel button, and the system will remove the car from the customers rental history, and send a notification of the purposed action. 5. They System will then release the car back into pool of available vehicles for future rentals.

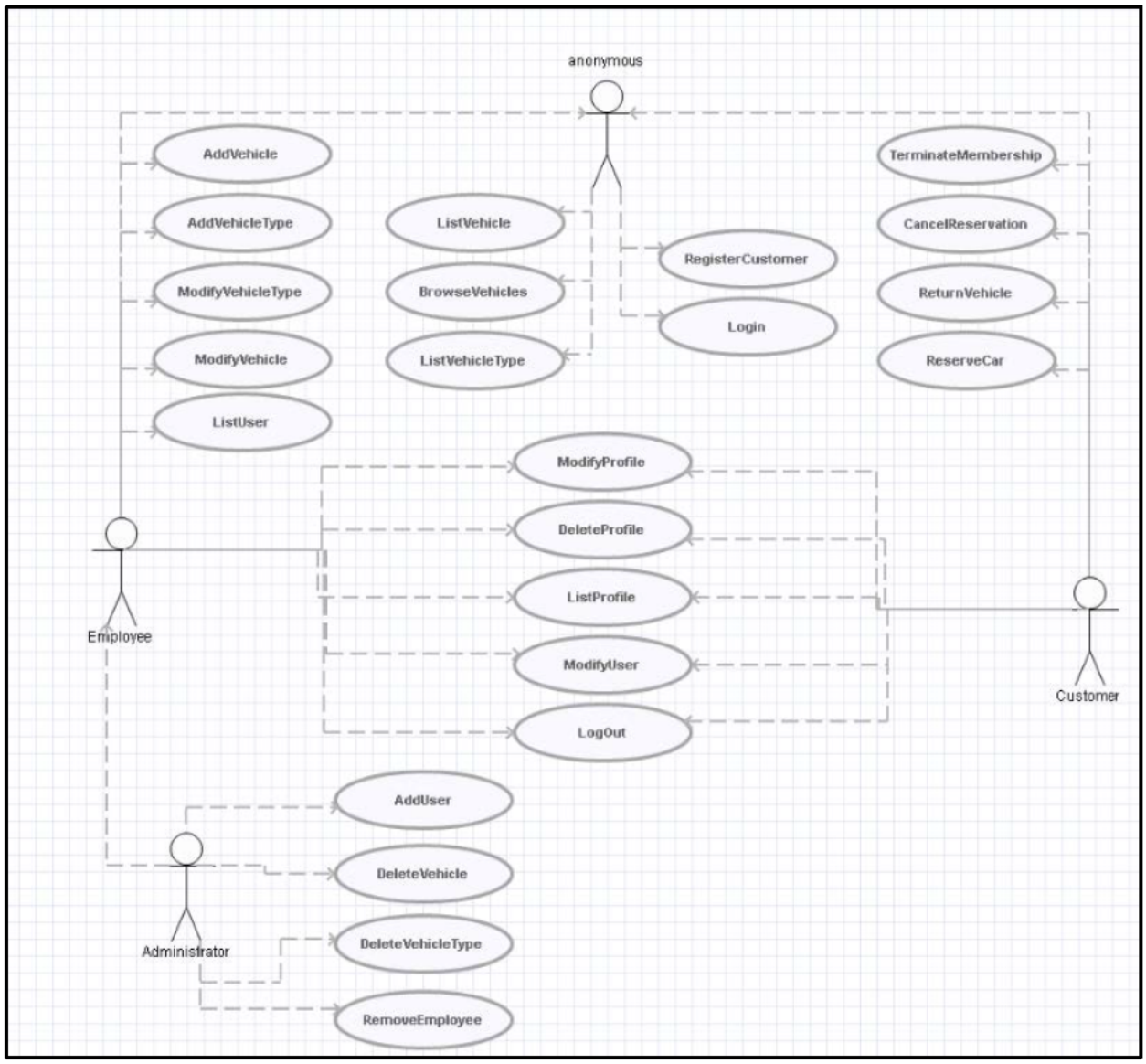
Scenario name	**terminateSubscription**
Participating actor Instances	Mike: Employee
Flow of events	1. Mike logs in to the car service rental system, and he wants to terminate a Customer Subscription. Mike will search for the customer by clicking the Search Customer button within the employee view. 2. The system will then display a search bar. Mike will then search the first and last name of the desired customer. 3. The system will then display the desired customer. Mike will then click on the desired customers information, and the System will display the Customers Information including car rental history. 4. Mike will then scroll to the button and click and options button followed by a subscriptions information tab. The system will display the customer subscription information. Mike will then scroll down to the bottom and click the termination option. 5. The System will then wipe all information associate with a consumer except basic personal information like name, email, and rental history for business assurance purposes.

Scenario name	**employeeSearchVehicle**
Participating actor Instances	Mike: Employee
Flow of events	1. Mike will login into the car rental service system, and he wants to search a vehicles information. Mike will search for vehicle by clicking the search vehicle inventory button within the employee view. 2. The system will then display a drop down with list with year, model, and make. 3. Mike will input a specific year, model, and make. The system will locate the specific year, model, and make of a vehicle as desired, and display a list of vehicles with the selected vehicle. The list will include an extensive list of information including year, model, make, vin number, and rental status. 4. Mike will then click on the vehicle from the list of vehicles. The System will then display a window showing the vehicle's full specifications, and added details. 5. Mike will then either continue to browse over the selected vehicle information or click the exit button at the top of the window.

Scenario name	**employeeModifyVehicle**
Participating actor Instances	Mike: Employee
Flow of events	1. Mike locates a vehicle through search. 2. Mike will then choose to modify the vehicle information by clicking the modify button. 3. The System will then reload a new page with forms next to the current information. Mike will then edit the fields of choice, and click the submit button at the bottom. 4. The system will then reload the information and display it to Mike and update the system. Mike will then click the modify button again to modify information or click the exit button at the top of the window to exit.

Scenario name	**employeeDeleteVehicle**
Participating actor Instances	Mike: Employee
Flow of events	1. Mike logs into the car rental service system and wants to modify the vehicle information. Mike will search and find the target vehicle. 2. Mike will then choose then delete vehicle option by scrolling down to the end of the selected vehicle information window and clicking the delete vehicle button. 3. The system will display a prompt window with a confirm button 4. Mike will click confirm. The system will remove all information about the vehicle from within the database.

2.4.2 Use Case Model



2.4.2.1 ReturnVehicle

Name	Return Vehicle
ID	ReturnVehicle
Version	1.0
Author	Stephen Patton
Date	9/8/13
Summary	Use case for customers notifying the system that they have returned the car they have rented
Basic Path	1. The system displays various information on the home page, including a link to the customer's account. The customer clicks on the link to his/her account 2. The system loads the account page. On the account page is a link that reads “Return Vehicle.” The customer clicks on the Return Vehicle. 3. The system prints a confirmation message asking if the customer is sure, and waits for the customer to confirm by clicking on the confirm button. 4. The customer clicks the confirm button and the system updates the cars being available. The system will then bring the customer back to the home page
Alternative Paths	1. In step 3, if the customer is late in returning the vehicle, the system will in addition to asking the user to confirm that the car is being returned, inform the customer of the additional money that is owed as a result of returning the car late. 2. In step 3, if the customer clicks cancel instead of confirm, he/she will be brought back to the account settings page.
Exception Paths	1. In step 2, the customer clicks on “Return Vehicle” when he/she has no vehicle currently being rented. The system will tell the customer that he/she has no vehicle currently rented and return to the account settings page
Extension Points	None
Triggers	A customer wants to return a vehicle
Assumption	None
Pre-conditions	The customer returns the car to the car rental site specified.
Post-conditions	The customer's payment is received, including possible late payments, and the car is updated to be available for other customers.

2.4.2.2 BrowseVehicles

Name	Browse Vehicles
ID	BrowseVehicles
Version	1.0
Author	Stephen Patton
Date	9/9/13
Summary	Use Case for browsing the Vehicles in the database through the website.
Basic Path	1. At the home page, a user clicks on the Browse button. 2. The system loads a page with options to limit browsing for a vehicle, including vehicle type, pick-up location, price range, vehicle make and model, and date and time available. 3. The user selects the limitations he/she wants and clicks on browse. 4. The system finds every vehicle in the database that falls within the search limitations and displays them on the screen
Alternative Paths	1. In step 4, if the user decides that he/she wants to narrow the search parameters, he/she can update them and the system will once again search the database and print the updated car list.
Exception Paths	1. If in step 4, the system can not find a single vehicle that matches the search, it will instead print that there are no vehicles that match that description. It will then give the user the option of making another search
Extension Points	None
Triggers	The user wants to browse available cars
Assumption	None
Pre-conditions	There are available cars
Post-conditions	The user can see the available cars within the parameters he/she has chosen.

2.4.2.3 RegisterCustomer

Name	Register Customer
ID	RegisterCustomer
Version	1.0
Author	Stephen Patton
Date	9/9/13
Summary	A new customer user registers with the system
Basic Path	1. The user clicks on the register a new account button 2. The system loads the registration page. There is a form asking for the following information: full name, phone number, address, email, driver's license information, and credit card information. 3. The user fills out the information and clicks continue 4. The system informs the user that there will be a membership fee and prompts the user to agree to pay. 5. The user clicks accept 6. The user is now registered with the system
Alternative Paths	1. In step 3, one or more of the information fields are incomplete or incorrect. The system asks the user to correct the information. The user corrects the information and moves on to step 4.
Exception Paths	1. In step 5, the user does not agree to pay the fee. The registration is terminated and the user is brought back to the home page
Extension Points	None
Triggers	A customer wants to register with the system
Assumption	The driver's license and credit card information are confirmed to be correct
Pre-conditions	None
Post-conditions	The database is updated to include this new user

2.4.2.4 AddVehicle

Name	Add Vehicle
ID	AddVehicle
Version	1.0
Author	Vincent Lee
Date	9/8/2013
Summary	Use case for an administrator/employee to add a vehicle to the fleet of existing vehicles.
Basic Path	1. The system provides a vehicles page listing the number of actions that the administrator/employee can take affecting the fleet of vehicles, after the administrator/employee is logged into the system. 2. The administrator/employee indicates that a new vehicle is to be added to the system. 3. The system prompts the administrator/employee for: a. The vehicle type of the new vehicle b. The make of the new vehicle c. The model of the new vehicle d. The year of the new vehicle e. The registration tag of the new vehicle f. The current mileage of the new vehicle g. The time of last servicing of the new vehicle h. The condition of the new vehicle i. The rental location assigned to the new vehicle The system also prompts the administrator/employee for action to take after the information is entered: a. Continue to review changes before saving to database b. Cancel current vehicle add 4. The administrator/employee clicks the save button in step 3, and the system displays a confirmation page with the entered information and prompts the administrator/employee to: a. Add Vehicle and return to the vehicle page b. Add Vehicle & Another c. Edit vehicle information d. Cancel current vehicle add 5. The administrator/employee chooses Add Vehicle and returns to the vehicle page, and the system stores the record for the new vehicle to the database.
Alternative Paths	1. In step 4, if the administrator/employee clicks the Edit button, the use case returns to step 3, with all of the information in the forms retained to allow necessary changes. 2. In step 3 and 4, if the administrator/employee clicks the Cancel button, the use case terminates, and the system returns to the vehicle page in step 1.
Exception Paths	1. In step 3, the system determines that a vehicle with the given

	registration tag already exists after Continue button is pressed, the system displays “A vehicle with registration tag *** already exists” message and the system stays at step 3. 2. In step 3, the system determines that a rental location is already at its maximum vehicle capacity after Continue button is pressed, the system displays “The rental location is at maximum capacity” message and the system stays at step 3.
Extension Points	1. In step 4, if the administrator/employee clicks Add Vehicle and Another button, the system adds the vehicle to the system, and returns to step 3 in the basic path.
Triggers	A user with the administrator/employee role is allowed to add a new vehicle to the system.
Assumption	None
Pre-conditions	The administrator/employee is currently logged in and the session has not expired.
Post-conditions	Information is appropriately added to the database of the system.

2.4.2.5 ListVehicle

Name	List Vehicle
ID	ListVehicle
Version	1.0
Author	Vincent Lee
Date	9/8/2013
Summary	Use case for listing all vehicles in the fleet of vehicles, given a set of conditions.
Basic Path	1. The system provides a search box which allows searching of vehicle information. 2. The system prompts a user to enter in a keyword to search the vehicles in the database. A Search button is located next to the search box. 3. The users fills in the search field and clicks the Search button. 4. The system searches through the database, and returns the result list to the user.
Alternative Paths	None
Exception Paths	1. If no vehicle are found, the system informs the user
Extension Points	None
Triggers	A user elects to list some kind of vehicle in the system.
Assumption	None
Pre-conditions	None
Post-conditions	The system displays the user with what he or she wants to view, and nothing changes to the underlying data in the system.

2.4.2.6 ModifyVehicle

Name	Modify Vehicle
ID	ModifyVehicle
Version	1.0
Author	Vincent Lee
Date	9/8/2013
Summary	Use case for an administrator/employee to modify the information of a vehicle.
Basic Path	1. The system provides a vehicles page listing the number of actions that the administrator/employee can take affecting the fleet of vehicles, after the administrator/employee is logged into the system. 2. The system implements the ListVehicle use case to locate the vehicle. 3. Once the vehicle is located, the administrator/employee clicks on the Edit button of the vehicle. 4. The system displays a vehicle profile page containing the information of the vehicle. 5. The administrator/employee makes the necessary modifications to the vehicle profile. The screen includes buttons: Continue and Cancel. 6. The administrator/employee clicks the Continue button and is directed to a confirmation page with the changes listed. The screen includes buttons: Save, Cancel, and Edit. 7. The administrator/employee clicks the Save button and changes are saved to the database.
Alternative Paths	1. In step 5 and 6, if the administrator/employee clicks the Cancel button, the use case terminates, and the system returns to the vehicle page in step 1.
Exception Paths	1. In step 6, if the administrator/employee clicks the Edit button, the use case returns to step 4, with all of the information in the forms retained to allow necessary changes.
Extension Points	None
Triggers	An administrator/employee elects to modify a vehicle’s information.
Assumption	None
Pre-conditions	The administrator/employee is currently logged in and the session has not expired.
Post-conditions	The modified information is available to other use cases. The attributes of the vehicle changed are appropriately updated in the database.

2.4.2.7 DeleteVehicle

Name	Delete Vehicle
ID	DeleteVehicle
Version	1.0
Author	Vincent Lee
Date	9/8/2013
Summary	Use case for an administrator/employee to remove a vehicle from the fleet of existing vehicles.
Basic Path	1. The system provides a vehicles page listing the number of actions that the administrator/employee can take affecting the fleet of vehicles, after the administrator/employee is logged into the system. 2. The system implements the ListVehicle use case to locate the vehicle. 3. Once the vehicle is located, the administrator/employee clicks on the Edit button of the vehicle. 4. The system displays a vehicle profile page containing the information of the vehicle. The screen contains the Delete button at the bottom. 5. The administrator/employee clicks the Delete button. 6. The system displays a confirmation screen containing the information of the vehicle to be deleted. The screen has buttons: Delete and Cancel. 7. An administrator/employee clicks the Delete button and the system displays a “Successfully deleted” message, and returns to the main vehicle page.
Alternative Paths	1. In step 6, if the administrator/employee clicks the Cancel button, the use case terminates, and the system returns to the vehicle page in step 1.
Exception Paths	None
Extension Points	None
Triggers	A user with administrator/employee role elects to delete a vehicle from the system.
Assumption	None
Pre-conditions	The administrator/employee is currently logged in and the session has not expired.
Post-conditions	The records concerning the vehicle the administrator wants to remove are deleted from the database.

2.4.2.8 AddVehicleType

Name	Add Vehicle Type
ID	AddVehicleType
Version	1.0
Author	Vincent Lee
Date	9/8/2013
Summary	Use case for an administrator/employee to add a vehicle type to the existing vehicles types available.
Basic Path	1. The system provides a vehicles page listing the number of actions that the administrator/employee can take affecting the fleet of vehicles, after the administrator/employee is logged into the system. 2. The administrator/employee indicates that a new vehicle type is to be added to the system. 3. The system prompts the administrator/employee for: a. The name of the new vehicle type b. The hourly rental price of the new vehicle type c. The daily rental price of the new vehicle type The system also prompts the administrator/employee for action to take after the information is entered: a. Continue to review changes before saving to database b. Cancel current vehicle type add 4. The administrator/employee clicks the save button in step 3, and the system displays a confirmation page with the entered information and prompts the administrator/employee to: a. Add Vehicle Type and return to the vehicle page b. Add Vehicle Type & Another c. Edit vehicle type information d. Cancel current vehicle type add 5. The administrator/employee chooses Add Vehicle Type and returns to the vehicle page, and the system stores the record for the new vehicle type to the database.
Alternative Paths	1. In step 4, if the administrator/employee clicks the Edit button, the use case returns to step 3, with all of the information in the forms retained to allow necessary changes. 2. In step 3 and 4, if the administrator/employee clicks the Cancel button, the use case terminates, and the system returns to the vehicle page in step 1.
Exception Paths	1. In step 3, the system determines that a vehicle type with the given name already exists after Continue button is pressed, the system displays “The vehicle type already exists” message and the system stays at step 3.
Extension Points	1. In step 4, if the administrator/employee clicks Add Vehicle Type and Another button, the system adds the vehicle type to the system, and returns to step 3 in the basic path.

Triggers	A user with the administrator/employee role is allowed to add a new vehicle type to the system.
Assumption	None
Pre-conditions	The administrator/employee is currently logged in and the session has not expired.
Post-conditions	Information is appropriately added to the database of the system.

2.4.2.9 ListVehicleType

Name	List Vehicle Type
ID	ListVehicleType
Version	1.0
Author	Vincent Lee
Date	9/8/2013
Summary	Use case for listing all vehicle types contained within the system, given a set of conditions.
Basic Path	1. The system provides a search box which allows searching of vehicle type. 2. The system prompts a user to enter in a keyword to search the vehicle types in the database. A Search button is located next to the search box. 3. The users fills in the search field and clicks the Search button. 4. The system searches through the database, and returns the result list to the user.
Alternative Paths	None
Exception Paths	None
Extension Points	None
Triggers	A user elects to list some kind of vehicle type in the system.
Assumption	None
Pre-conditions	None
Post-conditions	The system displays the user with what he or she wants to view, and nothing changes to the underlying data in the system.

2.4.2.10 ModifyVehicleType

Name	Modify Vehicle Type
ID	ModifyVehicleType
Version	1.0
Author	Vincent Lee
Date	9/8/2013
Summary	Use case for an administrator/employee to modify the information in a vehicle type.
Basic Path	1. The system provides a vehicles page listing the number of actions that the administrator/employee can take affecting the fleet of vehicles, after the administrator/employee is logged into the system. 2. The ListVehicleType use case is implemented. 3. Once the vehicle type is located, the administrator/employee clicks on the Edit button of the vehicle type. 4. The administrator/employee makes the necessary modifications to the vehicle type. The screen includes buttons: Continue and Cancel. 5. The administrator/employee clicks the Continue button and is directed to a confirmation page with the changes listed. The screen includes buttons: Save, Cancel, and Edit. 6. The administrator/employee clicks the Save button and changes are saved to the database.
Alternative Paths	1. In step 4 and 5, if the administrator/employee clicks the Cancel button, the use case terminates, and the system returns to the vehicle page in step 1.
Exception Paths	1. In step 5, if the administrator/employee clicks the Edit button, the use case returns to step 4, with all of the information in the forms retained to allow necessary changes.
Extension Points	None
Triggers	An administrator/employee elects to modify a vehicle type information.
Assumption	None
Pre-conditions	The administrator/employee is currently logged in and the session has not expired.
Post-conditions	The modified information is available to other use cases. The attributes of the vehicle type changed are appropriately updated in the database.

2.4.2.11 DeleteVehicleType

Name	Delete Vehicle Type
ID	DeleteVehicleType
Version	1.0
Author	Vincent Lee
Date	9/8/2013
Summary	Use case for an administrator/employee to delete a vehicle type from the list of vehicle types.
Basic Path	1. The system provides a vehicles page listing the number of actions that the administrator/employee can take affecting the fleet of vehicles, after the administrator/employee is logged into the system. 2. The system implements the ListVehicleType use case to locate the vehicle. 3. Once the vehicle type is located, the administrator/employee clicks on the Edit button of the vehicle type. 4. The system displays a vehicle type profile page containing the information of the vehicle type. The screen contains the Delete button at the bottom. 5. The administrator/employee clicks the Delete button. 6. The system displays a confirmation screen containing the information of the vehicle type to be deleted. The screen has buttons: Delete and Cancel. 7. An administrator/employee clicks the Delete button and the system displays a “Successfully deleted” message, and returns to the main vehicle page.
Alternative Paths	1. In step 6, if the administrator/employee clicks the Cancel button, the use case terminates, and the system returns to the vehicle page in step 1.
Exception Paths	None
Extension Points	None
Triggers	A user with administrator/employee role elects to delete a vehicle type from the system.
Assumption	None
Pre-conditions	The administrator/employee is currently logged in and the session has not expired.
Post-conditions	The records concerning the vehicle type the administrator wants to remove are deleted from the database.

2.4.2.12 ListProfile

Name	List Profile
ID	ListProfile
Version	1.0
Author	Vincent Lee
Date	9/8/2013
Summary	Use case for listing all user information in the system, given a set of conditions.
Basic Path	1. The system provides a search box which allows searching of profiles. 2. The system prompts a user to enter in a keyword to search the profiles in the database. A Search button is located next to the search box. 3. The users fills in the search field and clicks the Search button. 4. The system searches through the database, and returns the result list to the user.
Alternative Paths	None
Exception Paths	None
Extension Points	None
Triggers	A user elects to list some kind of profile in the system.
Assumption	Customers are only able to list their own profile.
Pre-conditions	None
Post-conditions	The system displays the user with what he or she wants to view, and nothing changes to the underlying data in the system.

2.4.2.13 ModifyProfile

Name	Modify Profile
ID	ModifyProfile
Version	1.0
Author	Vincent Lee
Date	9/8/2013
Summary	Use case for modifying the information contained in a user profile.
Basic Path	1. The system displays a user profile page listing the number of actions users can take based on their role. 2. The ListProfile use case is implemented. 3. Once the profile is located, the user clicks on the Edit button of the profile. 4. The user makes the necessary modifications to the profile. The screen includes buttons: Continue and Cancel. 5. The user clicks the Continue button and is directed to a confirmation page with the changes listed. The screen includes buttons: Save, Cancel, and Edit. 6. The user clicks the Save button and changes are saved to the database.
Alternative Paths	1. In step 4 and 5, if the user clicks the Cancel button, the use case terminates, and the system returns to the profile in step 1.
Exception Paths	1. In step 5, if the user clicks the Edit button, the use case returns to step 4, with all of the information in the forms retained to allow necessary changes.
Extension Points	None
Triggers	A user elects to modify a profile’s information.
Assumption	Customers are only able to modify their own profile.
Pre-conditions	The user is currently logged in and the session has not expired.
Post-conditions	The modified information is available to other use cases. The attributes of the profile changed are appropriately updated in the database.

2.4.2.14 DeleteProfile

Name	Delete Profile
ID	DeleteProfile
Version	1.0
Author	Vincent Lee
Date	9/8/2013
Summary	Use case for deleting a user profile from the list of user profiles.
Basic Path	1. The system displays a user profile page listing the number of actions users can take based on their role. 2. The system implements the ListProfile use case to locate the profile. 3. Once the profile is located, the user clicks on the Edit button of the user profile. 4. The system displays a profile page containing the information of the user. The screen contains the Delete button at the bottom. 5. The user clicks the Delete button. 6. The system displays a confirmation screen containing the information of the user profile to be deleted. The screen has buttons: Delete and Cancel. 7. A user clicks the Delete button and the system displays a “Successfully deleted” message, and returns to the main profile page.
Alternative Paths	1. In step 6, if the user clicks the Cancel button, the use case terminates, and the system returns to the profile page in step 1.
Exception Paths	None
Extension Points	None
Triggers	A user with elects to delete a profile from the system.
Assumption	Customers are only able to delete their own profile.
Pre-conditions	The user is currently logged in and the session has not expired.
Post-conditions	The records concerning the profile the user wants to remove are deleted from the database.

2.4.2.15 Login

Name	Login
ID	Login
Version	1.0
Author	Osama Mansour
Date	9/6/13
Summary	Use case to login into the registration systems; Necessary to gain access to functionalities of the system
Basic Path	1. The system will display a screen prompting for user name and passwords. The login button will be displayed as well 2. A user provides a username and password and the login button. 3. The system locates the user matching the username and password. The system displays a customized navigation view dependent on the user information (Employee, Administrator, Customer).
Alternative Paths	None
Exception Paths	1. In step 2, if the user name and password are not located in the database, it returns an invalid user message.
Extension Points	None
Triggers	A user chooses to use the system
Assumption	None
Pre-conditions	The user is not already logged in
Post-conditions	The user session is available to other user cases accessible to the same user

2.4.2.16 LogOut

Name	Log Out
ID	LogOut
Version	1.0
Author	Osama Mansour
Date	9/6/13
Summary	Use case for a user to logout from the car rental service system, terminating the current user's session.
Basic Path	1. A user presses logout button located in the user's view. 2. The system will terminate the current User's session and display the confirmation of the logout process
Alternative Paths	None
Exception Paths	None
Extension Points	None
Triggers	A user elects to quit using the system and logout
Assumption	None
Pre-conditions	The user is currently logged inside the system and desires to exit. The user session is also intact
Post-conditions	The current session is terminated for this user.

2.4.2.17 AddUser

Name	Add a new employee or administrator
ID	AddUser
Version	1.0
Author	Osama Mansour
Date	9/6/13
Summary	Use case for adding an employed user (administrator or employee) to the car rental system.
Basic Path	1. The system will provide the administrator with a list of actions that can be chosen on their initial display page. 2. The administrator indicates a new employed user is to be added to the system. 3. The system will then prompt the administrator for a **name**, **role**, **ID**, and **password.** The system will prompt for an option to submit to the database or cancel. 4. The administrator will then be prompted to return to the main page or add another user. 5. The administrator returns to main page and the system updates the new employed user into the database.
Alternative Paths	1. In step 3 the administrator could have clicked the cancel button terminating the use case, and returning to the main page.
Exception Paths	1. In step 3 the system determines that a user with the given ID is already exists, and a “User exists” message will display
Extension Points	1. In step 4 the user could choose to add another user, the systems receives the submission, and returns to the main page
Triggers	A user with admin/employee role elects to add a new user to the system
Assumption	The employee has not been suspended from adding users
Pre-conditions	The administrator is currently logged and the session is present.
Post-conditions	Information submitted is stored in the database correctly.

2.4.2.18 ListUser

Name	List User
ID	ListUser
Version	1.0
Author	Osama Mansour
Date	9/6/13
Summary	Use case for administrators and employees to list users according to search settings
Basic Path	1. The system provides a main page listing number of actions that the administrator can take after the administrator logs into the system. 2. The system prompts the administrator for the search option from the list of actions. 3. The system prompts the administrator with a drop down list of roles administrator, employee, or customer 4. The system then prompts for two options. a. Start Search option to list the users b. a field to add a last name to the Search bar c. Cancel button back to main page 5. The administrator clicks the Start search option to list the users 6. The system searches through the database and returns all the results bounded by the search settings given.
Alternative Paths	1. In step 4, the administrator clicks Cancel button, returning to main page
Exception Paths	None
Extension Points	None
Triggers	An administrator desires to search a particular user
Assumption	None
Pre-conditions	The administrator is logged into the system with a current session present.
Post-conditions	The system displays the administrator the information requested through the search settings.

2.4.2.19 ModifyUser

Name	Modify User
ID	ModifyUser
Version	1.0
Author	Osama Mansour
Date	9/6/13
Summary	User case for User to modify User information within constraints placed by the system. a. Administrator may edit themselves, employee, and customer information b. Employee may only edit Customer Information c. Customers may only edit themselves
Basic Path	1. The system provides a main page with a list of options one of which one option will allow user to modify his or her information. 2. The user will click the modify option and be directed to system display. 3. The user will then edit his or her information within the database by text field or drop box options designated by the system. 4. The user will hit the submit button. They system will display a confirmation window prompting the user to hit confirm or cancel updates. 5. The user will hit confirm. The system will locate the users information, and make the necessary updates to the database.
Alternative Paths	1. The admin/employee will have the modified permissions described in the summary of the use case. Dependent on the options chosen the Basic path from step 3 2. In step 4 the user hits cancel returning the system to the initial state.
Exception Paths	None
Extension Points	None
Triggers	A user wants to edit his/her information or other user information within the system.
Assumption	The user has the correct permissions granted
Pre-conditions	The user is currently logged in to the system and the session has not expired.
Post-conditions	The user has made the updates to the database of the system with accordance to the access rights defined.

2.4.2.20 RemoveEmployee

Name	Remove Employee
ID	RemoveEmployee
Version	1.0
Author	Osama Mansour
Date	9/6/13
Summary	Use case for administrator to remove an employee from the database.
Basic Path	1. The system provides a main page after login in listing the number of actions that can be performed by the administrator. 2. The administrator chooses the employee inventory option. The administrator then uses the list user use case to find the intended user to delete from the system. 3. The administrator will then choose the delete employee option only presented to administrative users. The system will prompt the administrator for to click a confirmation button or cancel update. 4. The administrator confirms the request, and the user will be removed from the database. 5. The administrator will then be prompts to continue to remove users form the database or exit 6. The administrator clicks exit returning to the main administrator page
Alternative Paths	1. In step 3, the administrator cancels updates which will not remove the user and return the administrator back to the main page.
Exception Paths	1. In step 3, if the search for the user fails then a message will display “Employee does not exists within the database.” and returns the administrator back to beginning of the basic path
Extension Points	1. In step 5, the administrator chooses to remove another user, returning the user to the beginning of the basic path
Triggers	A user with the administrative role intends to remove users from a database
Assumption	None
Pre-conditions	The administrator is currently logged in with the current session intact.
Post-conditions	The administrator has removed the requested employees from the database

2.4.2.21 TerminateMembership

Name	Terminate Membership
ID	TerminateMembership
Version	1.0
Author	Osama Mansour
Date	9/6/13
Summary	Use case for a customer to terminate their member ship from within the car rental service system.
Basic Path	1. The system provides a list of actions displayed on the main page of the customer view one of which includes accounting settings. 2. The customer will select the account settings button and be redirected to a screen with various account information including name, address, license, and activity status. Below the displayed information will be an account termination button which customer will click. 3. The system will then display a prompt confirming the initiation of the termination of the membership with two options. a. Confirm Termination of Account b. Cancel Termination of account 4. The customer will choose the termination of the account. The customer will then be automatically be given a message by the system that they have been removed from the car rental system. 5. The system will log the user out of the database, and terminate the session.
Alternative Paths	1. In step 4, choosing cancel termination of account will exit the user from the termination screen back to the main customer view page.
Exception Paths	None
Extension Points	None
Triggers	A user desires to terminate his or her account
Assumption	They are a customer not an employee or administrator these will go through different processes stated in a use case
Pre-conditions	The user is logged into the system with a current active session
Post-conditions	The user is removed from all databases including information about the customer. The customer car rental history is preserved for business preservation purposes, and the current session is done logging the user out of the web page back to the log in screen.

2.4.2.22 CancelReservation

Name	Cancel Reservation
ID	CancelReservation
Version	1.0
Author	Osama Mansour
Date	9/6/13
Summary	User case for canceling a car reservation through an employee from within the car rental system.
Basic Path	1. Upon the request of a customer to cancel a reservation. The employee will be presented with a section within the employee view called customers in which the employee will click. 2. The system will provide a display for customer inventory with a search bar on top. 3. The employee would search the customer by typing in the last name, first name order of the desired customer. The System scroll window will updates its result based on the name provided. 4. The employee would then click on the desired customer. The system will provide a new page listing all the customers information along with a vehicle history button. 5. The employee will click the rental history button**.** The employee would click on the desired vehicle rental with an in process status. 6. The System will then produce a message asking does the employee desire to cancel the following reservation in the process, and two buttons a Cancel reservation or exit button. 7. The employee would click cancel reservation. The system would then update the status of the vehicle rented to an available status. Releasing the vehicle to other customers for rentals, and the customer rental history would update the changes on the scroll window display. 8. The employee would then click the exit button at the bottom of the rental history window, and would be returned to the customers information page. The employee would perform click another exit button to end up back to the employee home view.
Alternative Paths	1. In step 6, if the employee were to click exit the system would return the employee to the customer car rental history page.
Exception Paths	1. In step 3, if the employee searches a nonexistent customer the system will return a blank.
Extension Points	1. In step 7, If the employee desired to another car reservation they would click on the next desired vehicle and click cancel reservation.
Triggers	A customer request to cancel a reservation through an employee.
Assumption	The Customer has placed a 72 hour notice for the cancellation of a vehicle reservation.
Pre-conditions	The employee has received the request, and is logged into the car rental service system.

Post-conditions	The employee has cancelled the following reservation(s) on the request of the customer. Releasing the Car back into the pool of rentals for that location.

2.4.2.23 ReserveCar

Name	Reserve Car
ID	ReserveCar
Version	1.0
Author	Osama Mansour
Date	9/6/13
Summary	Use case for customers reserving a car through the car rental service.
Basic Path	1. The system will display a customer view of the home page with a list of options. One which will include a rent a car option. 2. The System will then display a new page with a scroll pane, and a list of available cars. The customer will click on his or her desired vehicle. 3. The system will display a new page with all the vehicle information and a button to rent a car. The customer would click on rent a car. 4. The system will update the database to store the car within the customers rental history, and remove the car for the pool of available cars, and return the user back to the customer home page. Charges will apply after the vehicle has been returned
Alternative Paths	1. In step 4, if the customer clicked cancel then he would be returned to the list of vehicles available.
Exception Paths	1. In step 2, if the system has no available cars it will display a screen with a message stating no cars are available. 2. In step 6, if the system has returned that the card was invalid. A message will appear stating that the credit card information was invalid.
Extension Points	1. In step 6, if vehicle is returned late then late fees will apply
Triggers	A customer wants to rent a vehicle.
Assumption	The Customer has not reached the max amount of rentals within one period
Pre-conditions	The customer is logged into the system and desires to rent a car.
Post-conditions	The customers has made his payment, and the car has been reserved for rental, and removed from the pool of available cars within the database.

2.4.3 Object Model

2.4.3.1 Data Dictionary

2.4.3.1.1 Entity Package

Class Name	User		
Purpose	The generalize entity involved in a Car rental system transaction		
SuperClass	None		
SubClass	Administrator, Employee, Customer		
Attributes	Name	Type	Description
	UserID	String	The ID of the User as defined by the System Registration
	password	String	The encryption on the user profile to be accessed
	role	String	The role of the user defined by the system
Operations	Name	Inputs	Description
	Login	UserID, Password role	The login page allows users access to registered users within a system
	Logout		The logout page allows users currently login in to exit there current system profile
	updateProfile		The update profile operation allows for user to change and save their profile information
Relationships	Super class of Administrator, Employee, Customer		

Class Name	DriversLicense		
Purpose	Provide driver’s license information of Customer		
SuperClass	None		
SubClass	None		
Attributes	Name	Type	Description
	number	String	The number on the license
	expiration	String	The expiration date
	dateOfBirth	String	The date of birth on the license
Operations	Name	Inputs	Description
Relationships	None		

Class Name	CreditCard		
Purpose	Provide credit card information of Customer		
SuperClass	None		
SubClass	None		
Attributes	Name	Type	Description
	number	String	The number on the card
	expiration	String	The expiration date
	code	Int	The CVC code
Operations	Name	Inputs	Description
Relationships	None		

Class Name	Customer			
Purpose	The generalization of group of Users labeled Customers			
SuperClass	Users			
SubClass	None			
Attributes	Name	Type	Description	
	Social Security Number	String	Customer social security information for verification	
	firstName	String	Customer First name	
	lastName	String	Customer Last name	
	address	String	Customer Address	
	telephone	String	Customer Telephone number	
Operations	Name	Inputs	Outputs	Description
	ViewMyInfo		Personal Information Display	Displays all personal information of the current logged in Customer
	ViewAvailbileCars	CarName CarLocation	Available Cars	Displays the Cars available for rental by location
	ViewMyRentalHistory		Rental History Report	Displays a current logged Customer rental history
	ViewMyBillingInfo		Billing Report	Displays Current logged Customer billing report
	ViewMyCarRental		Car Rental Report	Displays Customer past and present car rentals
	EditMyInfo	FirstName LastName Address Telephone #	Updated Personal Information	Allows Customers to edit their information as needed
Relationships	Subclass of the User A Customer can only rent 1 Car at a time A Customer can only edit personal information, but not vehicle information			

Class Name	Employee			
Purpose	The generalization of group of Users labeled Employee within the Car rental System			
SuperClass	Users			
SubClass	None			
Attributes	Name	Type	Description	
	EmployeeID	Int	ID identifying the employee within the car rental system	
	firstName	String	Employee First name	
	lastName	String	Employee Last name	
	address	String	Employee Address	
	telephone	String	Employee Telephone number	
Operations	Name	Inputs	Outputs	Description
	ViewMyInfo		Personal Information Display	Displays all personal information of the current logged in employee
	ViewVehicles	CarName CarLocation	Available Cars including model, make year, vin number, and repair history reports.	Displays the Cars available within the Car rental system
	ViewRentalHistory	Customer Name	Rental History Report on any Customer within the system	Displays a rental history for all customer by name
	ViewBillingHistory	Customer Name	Billing Report	Displays all Customer History Billing reports
	ViewCarRentals	Customer Name	Car Rental Report	Displays All Customer Car rental reports
	EditVehicleInfo	Vehicle Name	Updated Vehicle Information	Edit Vehicle Information
	EditCustomerInfo	FirstName	Update	Allows Employee to edit

		LastName Social Security Number	Customer Personal Information	customer information as long as they have the following information
Relationships	Subclass of the User An Employee can edit Vehicle Information, and Customer information on Request			

Class Name	Administrator			
Purpose	The generalization of group of Users labeled Administrator within the Car rental System			
SuperClass	Users			
SubClass	None			
Attributes	Name	Type	Description	
	AdminID	Int	ID identifying the Administrator within the car rental system	
	firstName	String	Employee First name	
	lastName	String	Employee Last name	
	address	String	Employee Address	
	telephone	String	Employee Telephone number	
Operations	Name	Inputs	Outputs	Description
	ViewMyInfo		Personal Information Display	Displays all personal information of the current logged in employee
	ViewVehicles	CarName CarLocation	Available Cars including model, make year, vin number, and repair history reports.	Displays the Cars available within the Car rental system
	ViewRentalHistory	Customer Name	Rental History Report on any Customer within the system	Displays a rental history for all customer by name
	ViewBillingHistory	Customer Name	Billing Report	Displays all Customer History Billing reports
	ViewCarRentals	Customer Name	Car Rental Report	Displays All Customer Car rental reports
	EditVehicleInfo	Vehicle Name	Updated Vehicle Information	Edit Vehicle Information
	EditCustomerInfo	CustomerName	Update	Allows Admin to edit

			Customer Personal Information	customer information both adding and removing Customers
	EditEmployeeInfo	EmployeeName	Updates Employee Information within the system	Allows Admin to edit Employee information within the system both adding and removing employees
Relationships	Subclass of the User An Employee can edit Vehicle Information, and Customer information on Request			

Class Name	VehicleType			
Purpose	An entity class for storing information on vehicle types			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
	name	String	The name of the vehicle type	
	hourly_rate	Double	The hourly price of the vehicle type	
	daily_rate	Double	The daily price of the vehicle type	
Operations	Name	Inputs	Outputs	Description
Relationships	None			

Class Name	Vehicle			
Purpose	An entity class for storing information on vehicles in the fleet			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
	make	String	The make of the vehicle	
	model	String	The model of the vehicle	
	year	Integer	The manufacture year of the vehicle	
	tag	String	The registration tag of the vehicle	
	mileage	Integer	The current mileage of the vehicle	
	last_serviced	Date	The last date the vehicle was serviced	
	condition	String	The condition of the vehicle	
Operations	Name	Inputs	Outputs	Description
Relationships	A vehicle belongs to 1 location			

Class Name	Location			
Purpose	An entity class for storing information on point of presence POP locations			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
	name	String	The name of the location	
	address	String	The address of the location	
	vehicle_capacity	Integer	The capacity of the location	
	vehicles	ArrayList<Vehicle>	The list of vehicles assigned to the location	
Operations	Name	Inputs	Outputs	Description
Relationships	None			

Class Name	Membership			
Purpose	An entity class for storing information on membership tiers			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
	name	String	The name of the membership tier	
	monthly_price	Double	The price of one month's service	
	duration	Integer	The length of membership in months	
Operations	Name	Inputs	Outputs	Description
Relationships	A customer belongs to 1 membership			

2.4.3.1.2 user.control package

Class Name	User			
Purpose	This control class allows a User to complete actions associated with login and profile modification			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	login	username password		Logs the user in with the appropriate permissions
	logout			Logs the user out
	loadProfile	username		Gets all relevant information related to the username
	updateProfile	profile	Boolean	Updates profile information and returns confirmation with Boolean
	registerCustomer	username password address creditcard driverlicense	Boolean	Allows a User to register as a customer
	vehicleSearch	name		Performs a search of vehicles in the fleet
Relationships	None			

2.4.3.1.3 user.boundary package

Class Name	LoginUI			
Purpose	This boundary class allows a User to log in			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	login	username password		Verify the user and logs the user with appropriate permissions
Relationships	None			

Class Name	LogoutUI			
Purpose	This boundary class allows a User to log out			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	logout			Logs the user out
Relationships	None			

Class Name	ChangeProfileUI			
Purpose	This boundary class allows a User to perform profile changes			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	updateProfile	profile		Updates the profile of a user
	saveProfile	profile	Boolean	Saves changes of profile change to data structure
Relationships	None			

Class Name	RegisterUI			
Purpose	This boundary class allows a User to register as a customer			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	registerCustomer	username password address creditcard driverlicense	Boolean	Registers a customer
Relationships	None			

Class Name	SearchUI			
Purpose	This boundary class allows a User to search the vehicle fleet			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	serach	keyword	List	Gets a list of available vehicles based on keyword
Relationships	None			

2.4.3.1.4 administrator.control package

Class Name	UserAdd			
Purpose	This control class allows Administrator to add users of all permissions			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	addUser	username password	Boolean	Adds a user to the database with Boolean confirmation
Relationships	None			

Class Name	VehicleRemove			
Purpose	This control class allows Administrator to remove a vehicle from the system			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	removeVehicle	vehicleProfile	Boolean	Removes a vehicle from the fleet with Boolean conformation
Relationships	None			

Class Name	VehicleTypeRemove			
Purpose	This control class allows a Administrator to remove a vehicle type from the system			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	removeVehicleType	vehicleType	Boolean	Removes a vehicletype from the database with confirmation
Relationships	None			

Class Name	EmployeeRemove			
Purpose	This control class allows a Adminstrator to remove an employee			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	removeEmployee	userProfile	Boolean	Remove an employee from the system with Boolean confirmation
Relationships	None			

2.4.3.1.5 administrator.boundary package

Class Name	UserAddUI			
Purpose	This boundary class allows an Administrator to add a user to the domain			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	addUser	username password	Boolean	Adds a user to the system confirmation
Relationships	None			

Class Name	RemoveVehicleUI			
Purpose	This boundary class allows Administrator to remove a vehicle from the fleet			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	removeVehicle	vehicleProfile	Boolean	Removes a vehicle from the fleet with conformation
Relationships	None			

Class Name	RemoveVehicleTypeUI			
Purpose	This boundary class allows Administrator to remove a vehicletype from the system			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	removeVehicleType	vehicleType	Boolean	Removes a vehicletype from the site with confirmation
Relationships	None			

Class Name	RemoveEmployeeUI			
Purpose	This boundary class allows Administrator to remove a employee from the system			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	removeEmployee	userProfile	Boolean	Remove an employee from the

				system with confirmation
Relationships	None			

2.4.3.1.6 employee.control package

Class Name	Vehicle			
Purpose	This control class allows Employee to modify vehicle information			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	addVehicle	vehicleType make model year tag mileage last_serviced condition	Boolean	Adds a vehicle to the fleet with Boolean confirmation
	modifyVehicle	vehicleProfile	Boolean	Modify's vehicle information with Boolean confirmation
Relationships	None			

Class Name	VehicleType			
Purpose	This control class allows Employee to modify vehicle type information			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	addVehicleType	name hourly_rate daily_rate	Boolean	Adds a vehicle type to the system with Boolean confirmation
	modifyVehicleType	vehicleType	Boolean	Modify's a vehicle type information with Boolean confirmation
Relationships	None			

Class Name	UserList			
Purpose	This control class allows Employee view customer's information			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	listUser	keyword	List	Returns a list of users based on keyword
Relationships	None			

2.4.3.1.7 employee.boundary package

Class Name	VehicleUI			
Purpose	This boundary class allows an Employee to perform actions on the fleet of vehicles			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	addVehicle	vehicleType make model year tag mileage last_serviced condition	Boolean	Adds vehicle to database with confirmation
	modifyVehicle	vehicleProfile	Boolean	Modifies vehicle information in the database with confirmation
Relationships	None			

Class Name	VehicleTypeUI			
Purpose	This boundary class allows a Employee to perform actions on the vehicle types			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	addVehicleType	name hourly_rate daily_rate	Boolean	Adds a vehicle type to the system with Boolean confirmation
	modifyVehicleType	vehicleType	Boolean	Modifies a vehicle type information with Boolean confirmation
Relationships	None			

Class Name	UserUI			
Purpose	This boundary class allows an Employee to view information on users			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	listUser	keyword	List	Lists users based on keyword match
Relationships	None			

2.4.3.1.8 customer.control package

Class Name	Membership			
Purpose	This control class allows a Customer to perform actions to their membership options			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	terminateMembership		Boolean	Terminates a membership of a customer with Boolean confirmation
Relationships	None			

Class Name	VehicleReservation			
Purpose	This control class allows a Customer to preform actions related to renting a vehicle			
SuperClass	None			
SubClass	None			
Attributes	Name	Type		Description
Operations	Name	Inputs	Outputs	Description
	reserveVehicle	vehicleProfile	Boolean	Reserves a vehicle with Boolean confirmation
	returnVehicle	vehicleProfile	Boolean	Checks a vehicle back into the fleet of available vehicles with Boolean confirmation
Relationships	None			

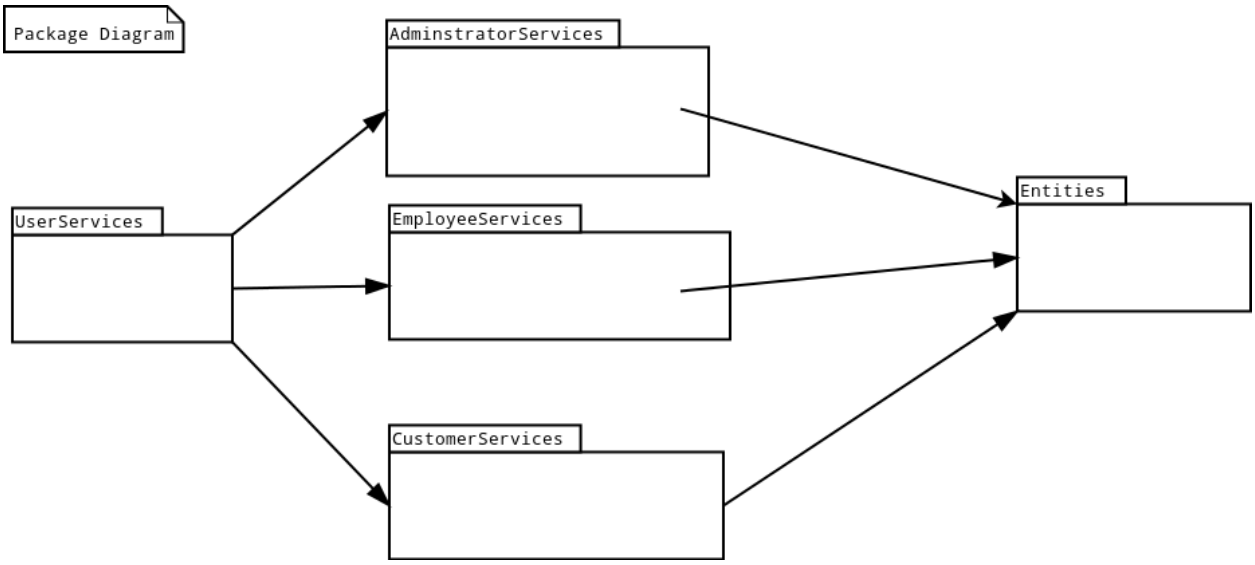
2.4.3.1.9 customer.boundary package

Class Name	MembershipUI			
Purpose	This boundary class allows a Customer to terminate their membership			
SuperClass	None			
SubClass	None			
Attributes	Name	Type		Description
Operations	Name	Inputs	Outputs	Description
	terminateMembership		Boolean	Terminates a customer's membership with conformation
Relationships	None			

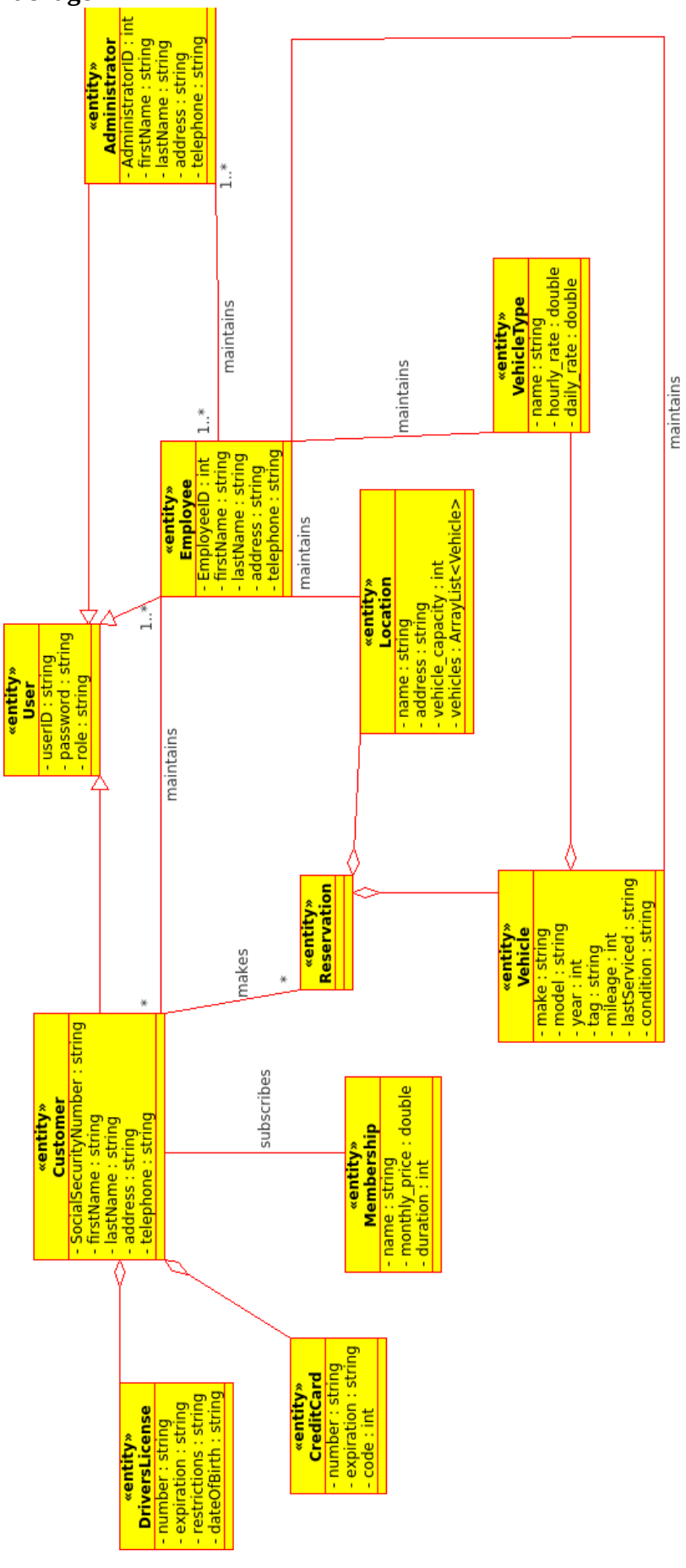
Class Name	ReservationUI			
Purpose	This boundary class allows a Customer to reserve a vehicle			
SuperClass	None			
SubClass	None			
Attributes	Name	Type	Description	
Operations	Name	Inputs	Outputs	Description
	reserveVehicle	vehicleProfile	Boolean	Reserves a vehicle with Boolean confirmation
	returnVehicle	vehicleProfile	Boolean	Checks a vehicle back into the fleet of available vehicles with Boolean confirmation
Relationships	None			

2.4.3.2 Class Diagrams

2.4.3.2.1 Package Diagram

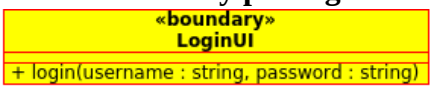


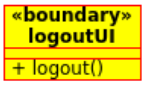
2.4.3.2.2 Entity Package

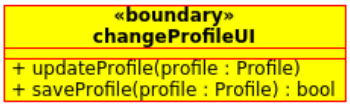


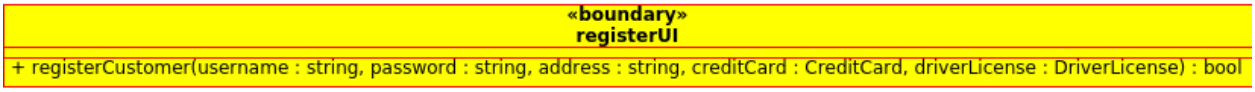
2.4.3.2.3 user.control package

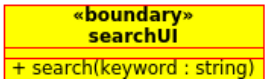
2.4.3.2.4 user.boundary package



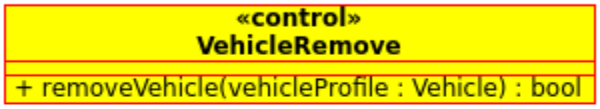


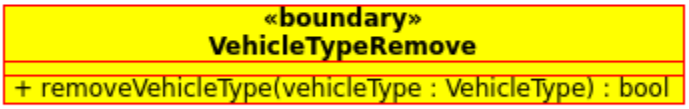


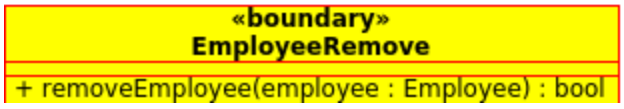




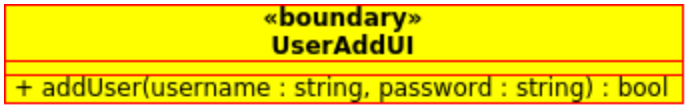
2.4.3.2.5 administrator.control package

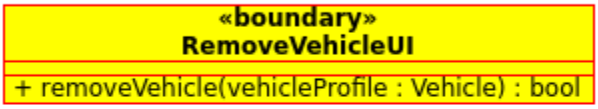


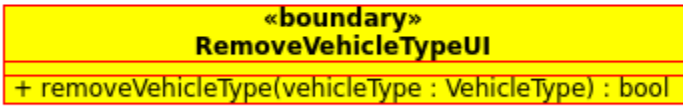


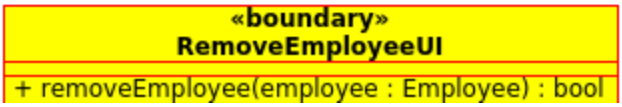


2.4.3.2.6 administrator.boundary package

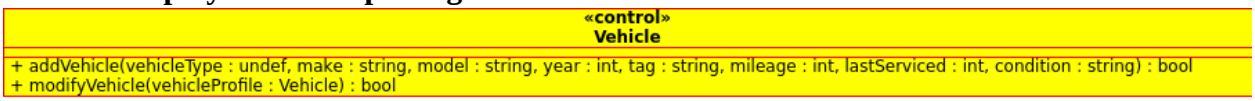


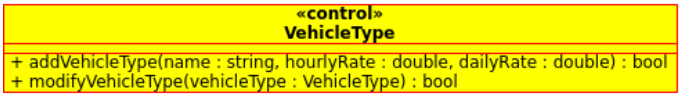




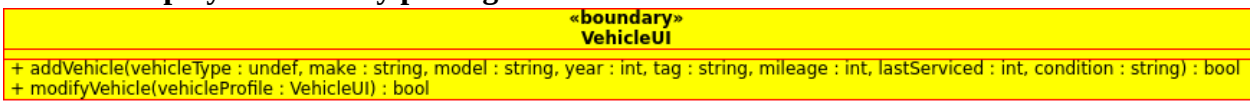


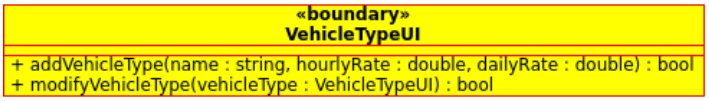
2.4.3.2.7 employee.control package





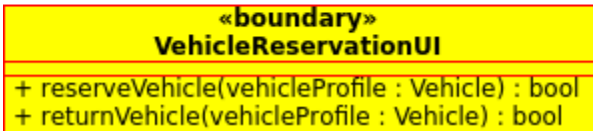
2.4.3.2.8 employee.boundary package





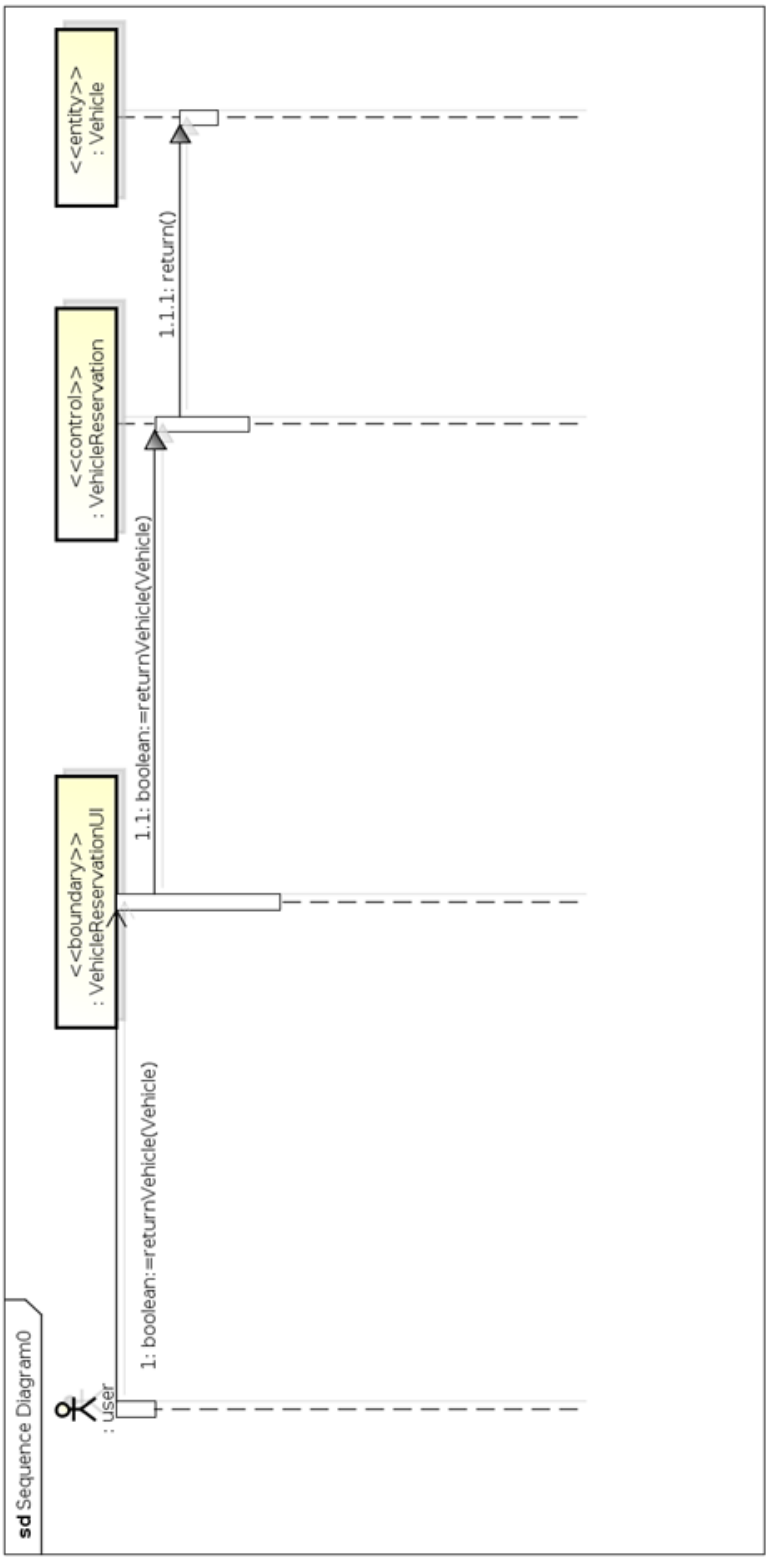
2.4.3.2.9 customer.control package

2.4.3.2.10 customer.boundary package

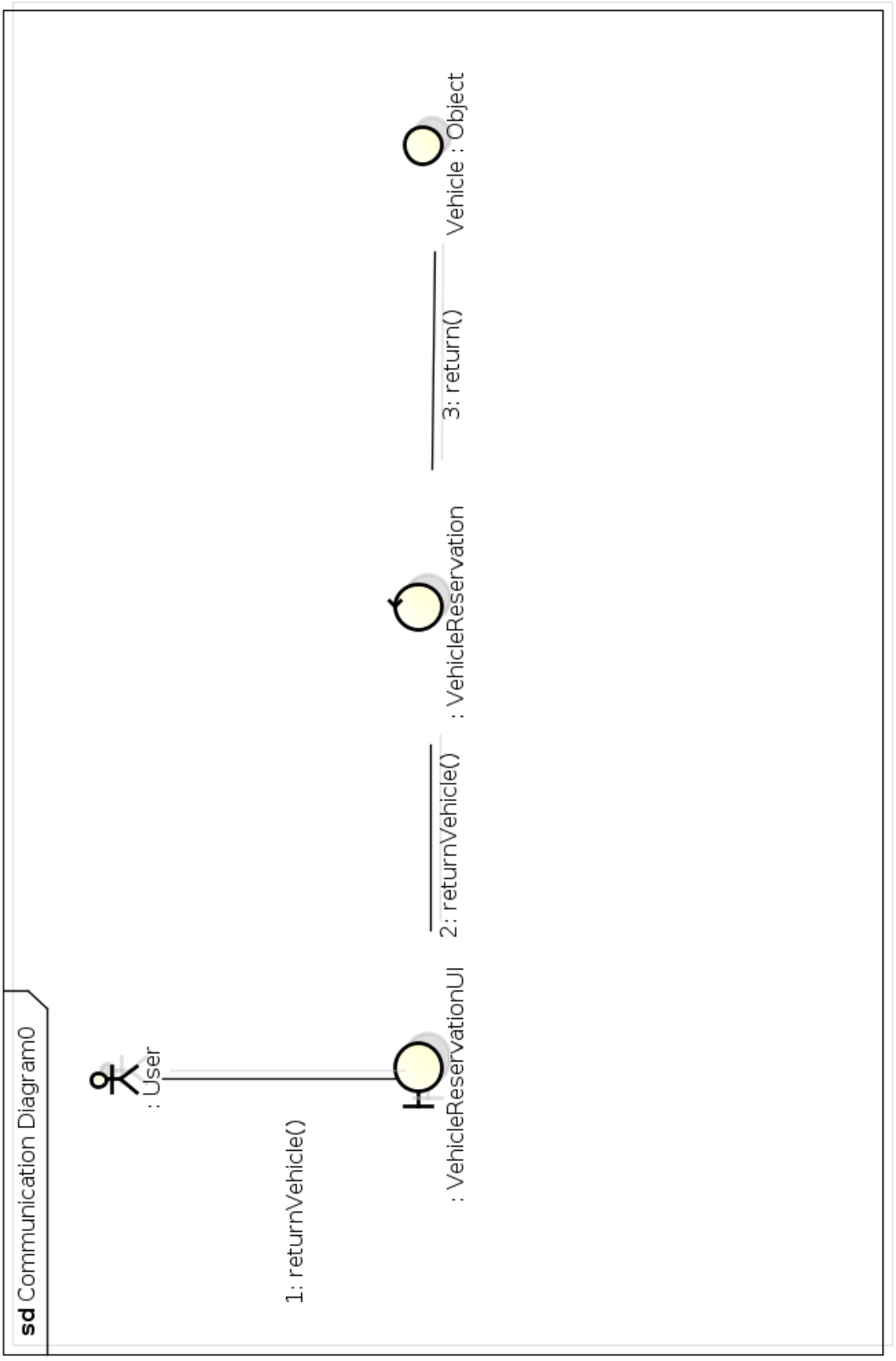


2.4.4 Dynamic Model – Sequence & Communication Diagrams

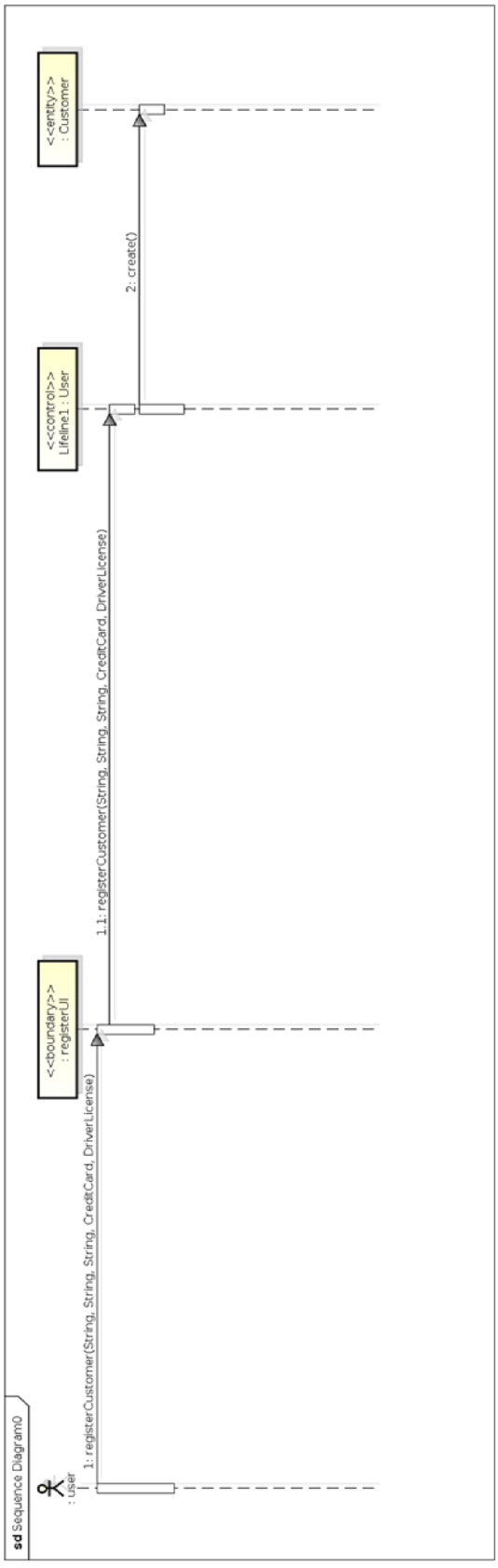
2.4.4.1 ReturnVehicle – Sequence Diagram – Use Case: Page 22



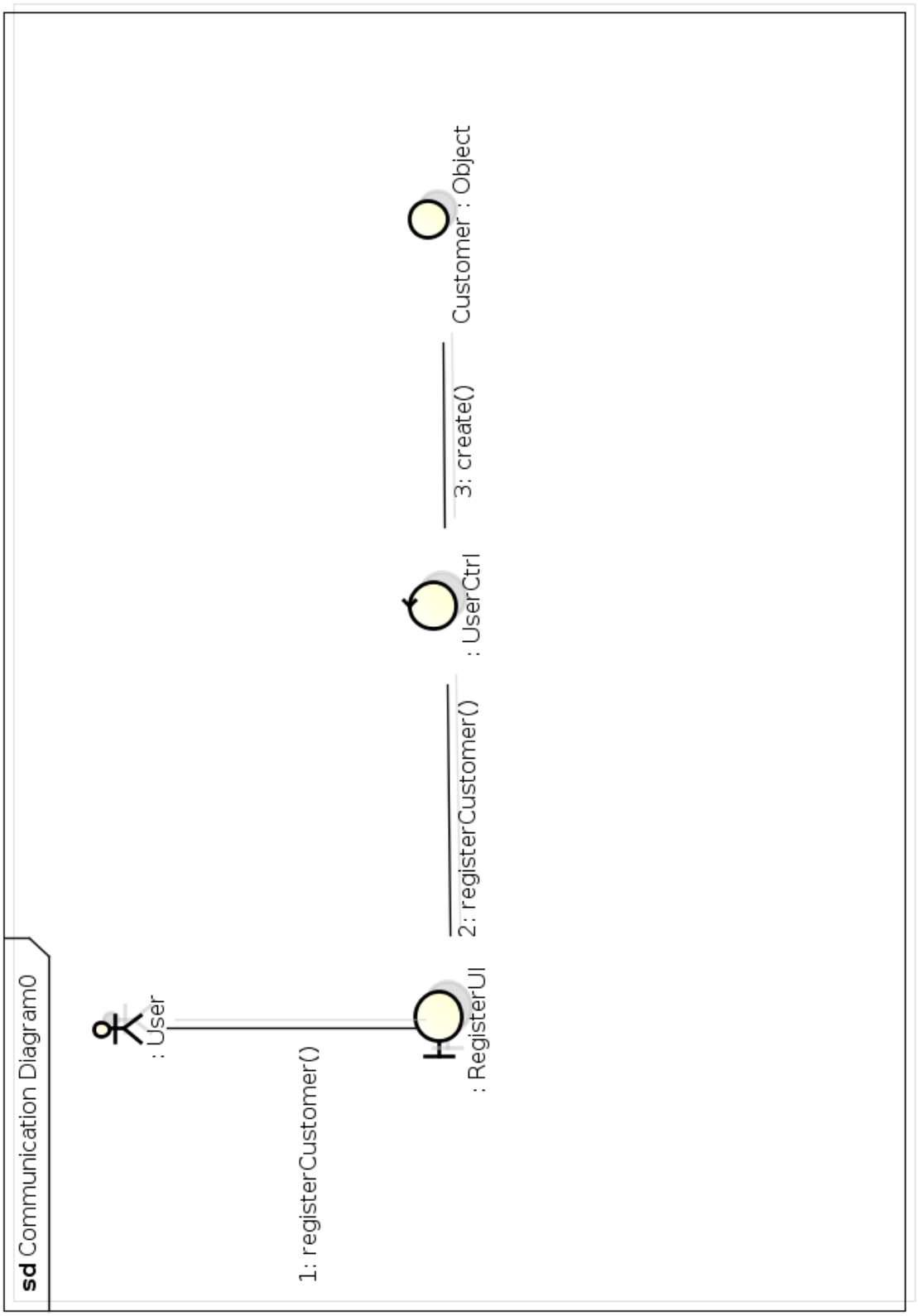
2.4.4.1 ReturnVehicle – Communication Diagram – Use Case: Page 22



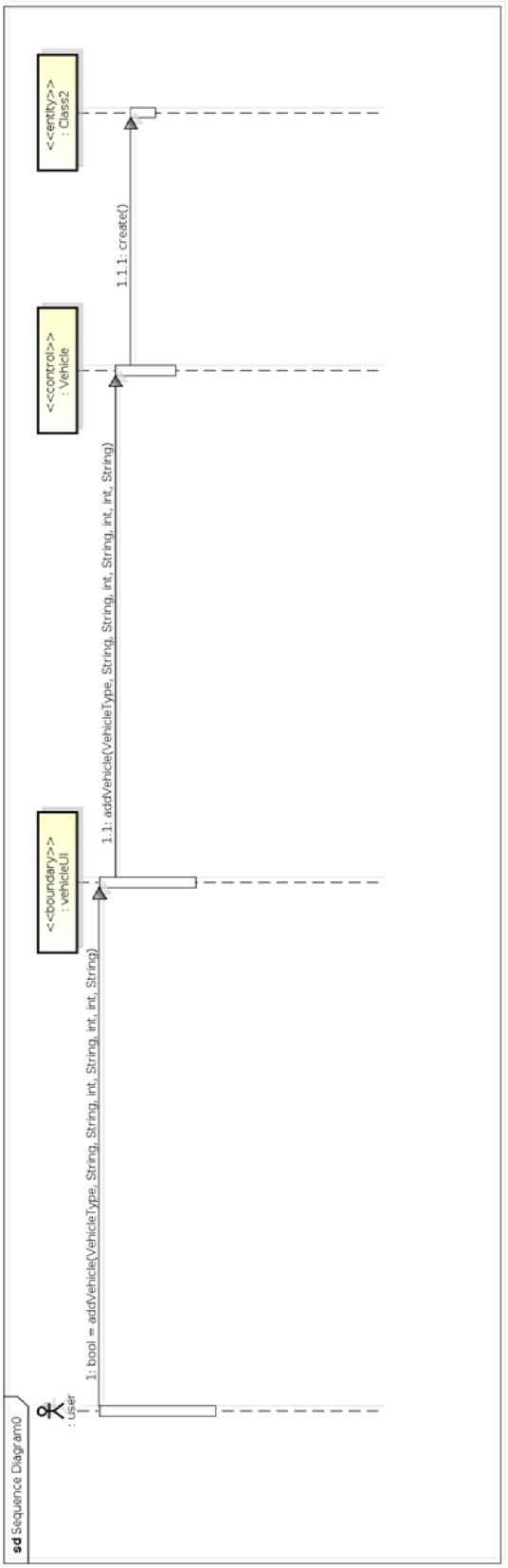
2.4.4.2 RegisterCustomer – Sequence Diagram – Use Case: Page 24



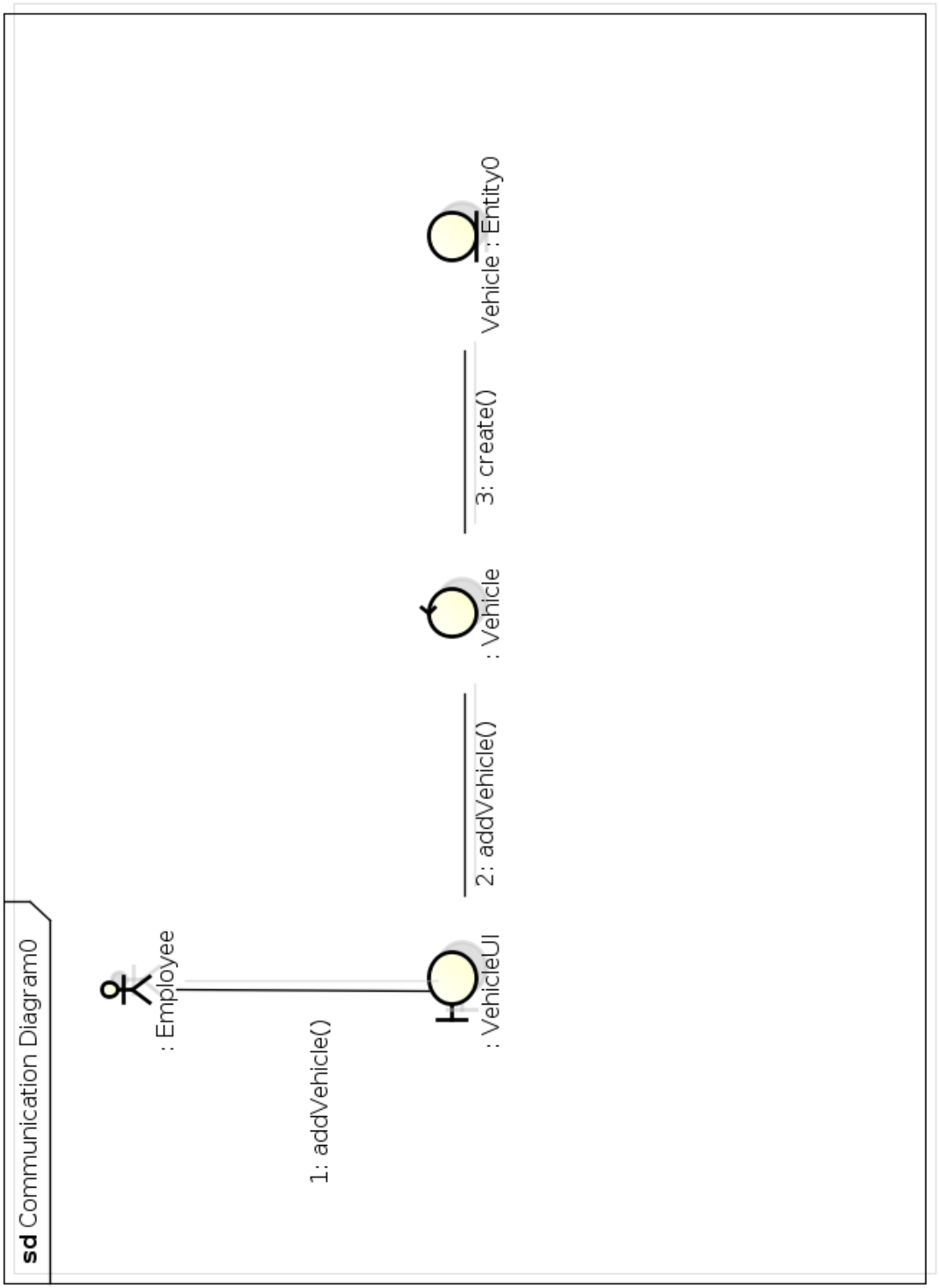
2.4.4.2 RegisterCustomer – Communication Diagram – Use Case: Page 24



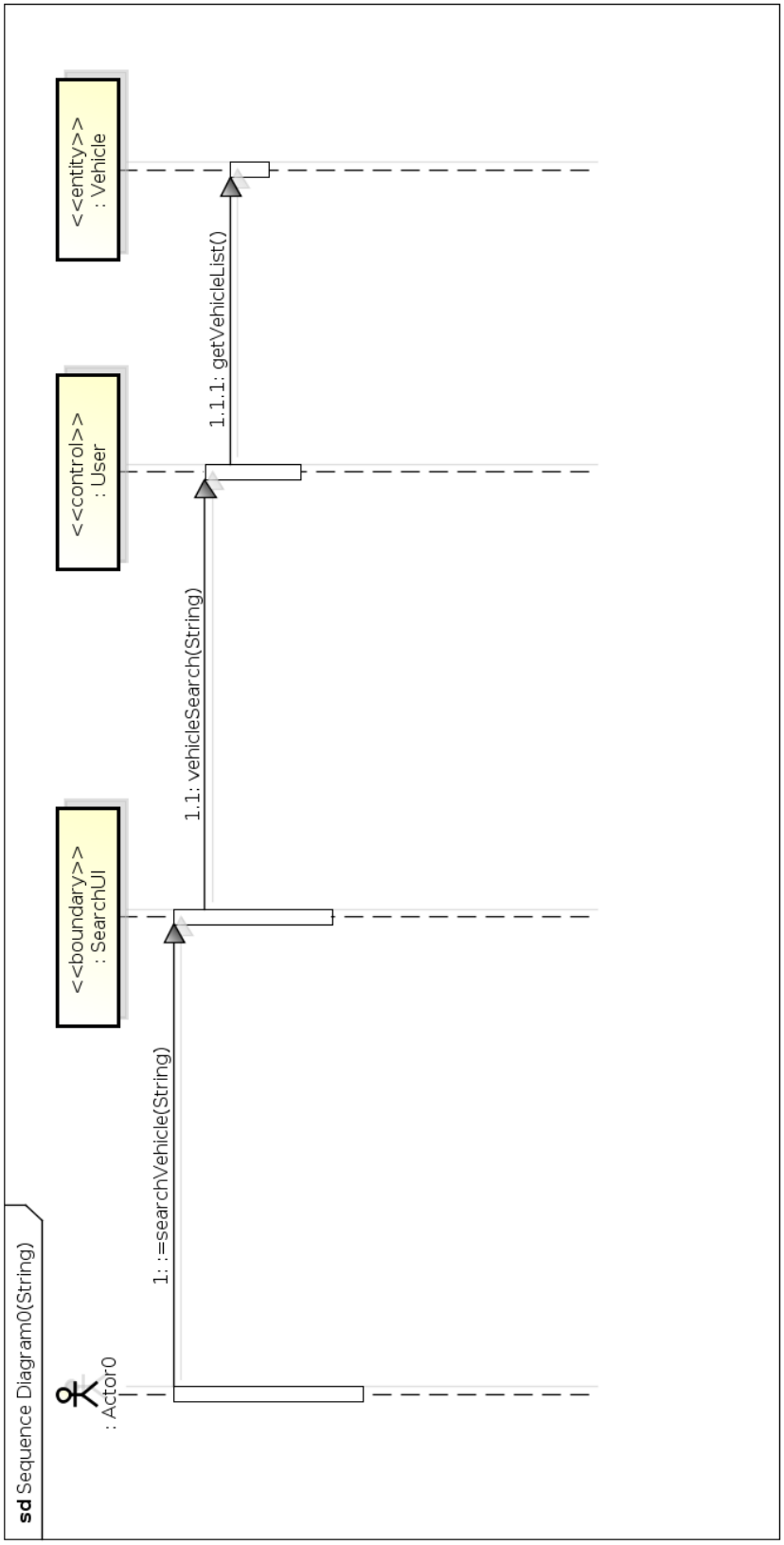
2.4.4.3 AddVehicle – Sequence Diagram – Use Case: Page 25



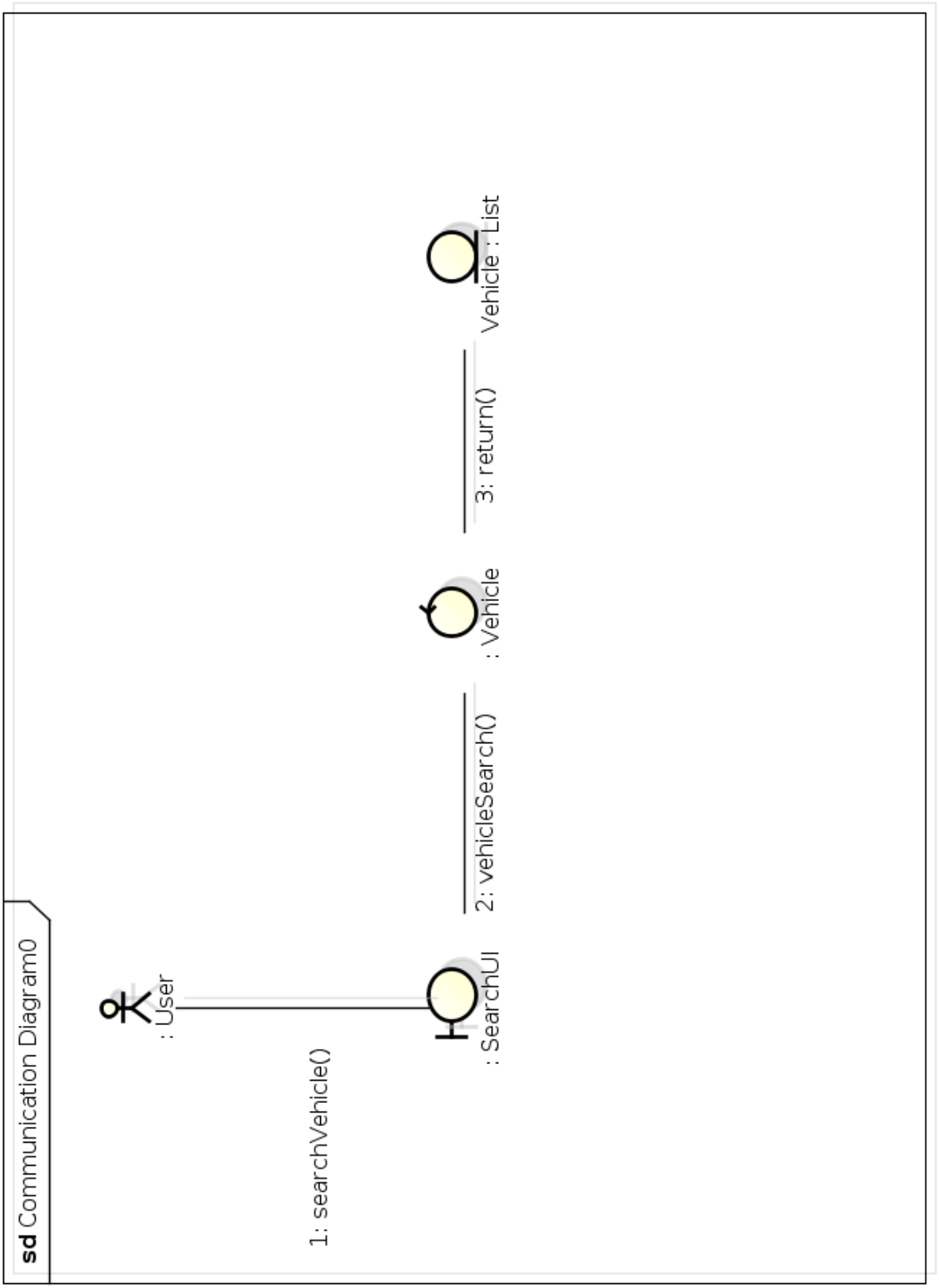
2.4.4.3 AddVehicle – Communication Diagram – Use Case: Page 25



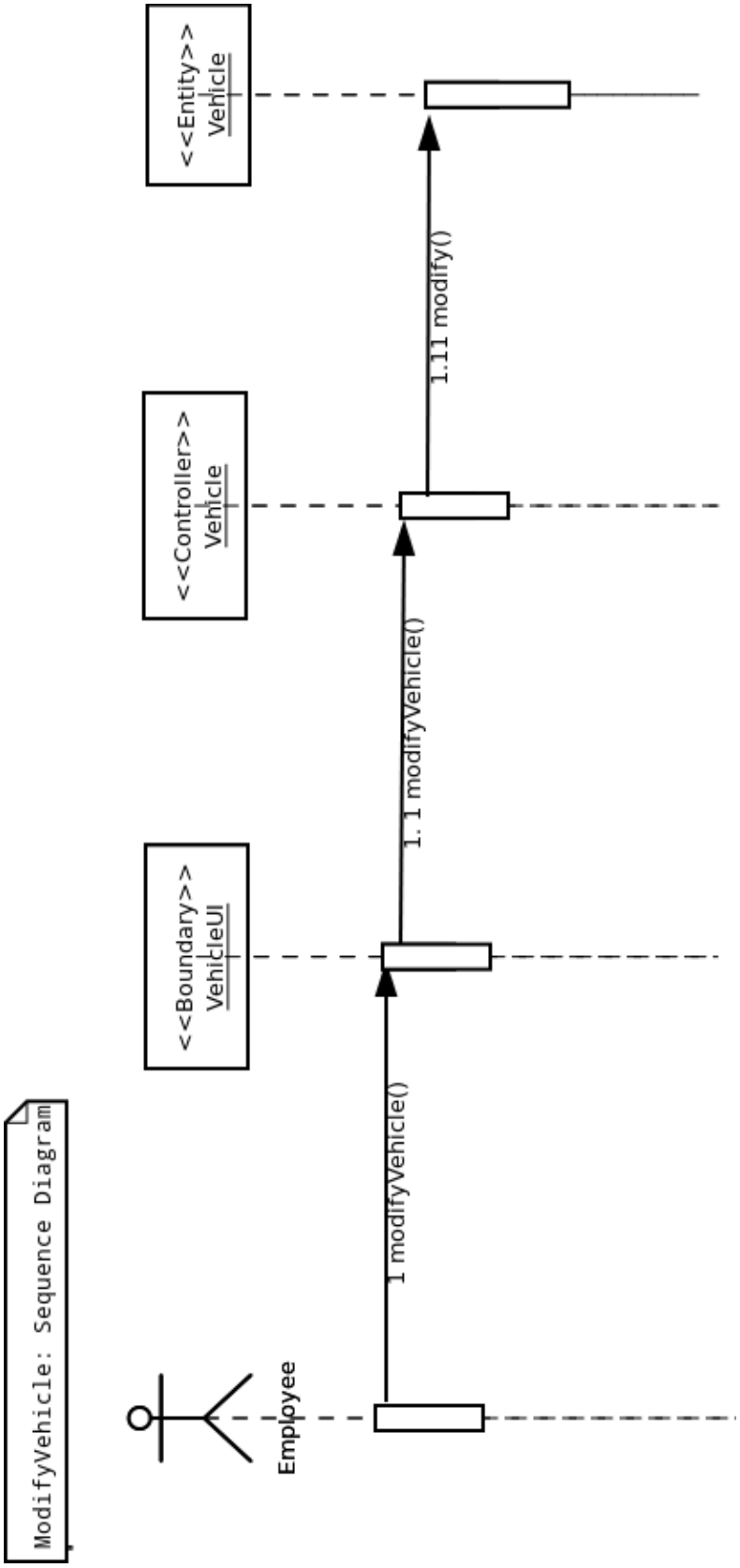
2.4.4.4 ListVehicle – Sequence Diagram – Use Case: Page 27



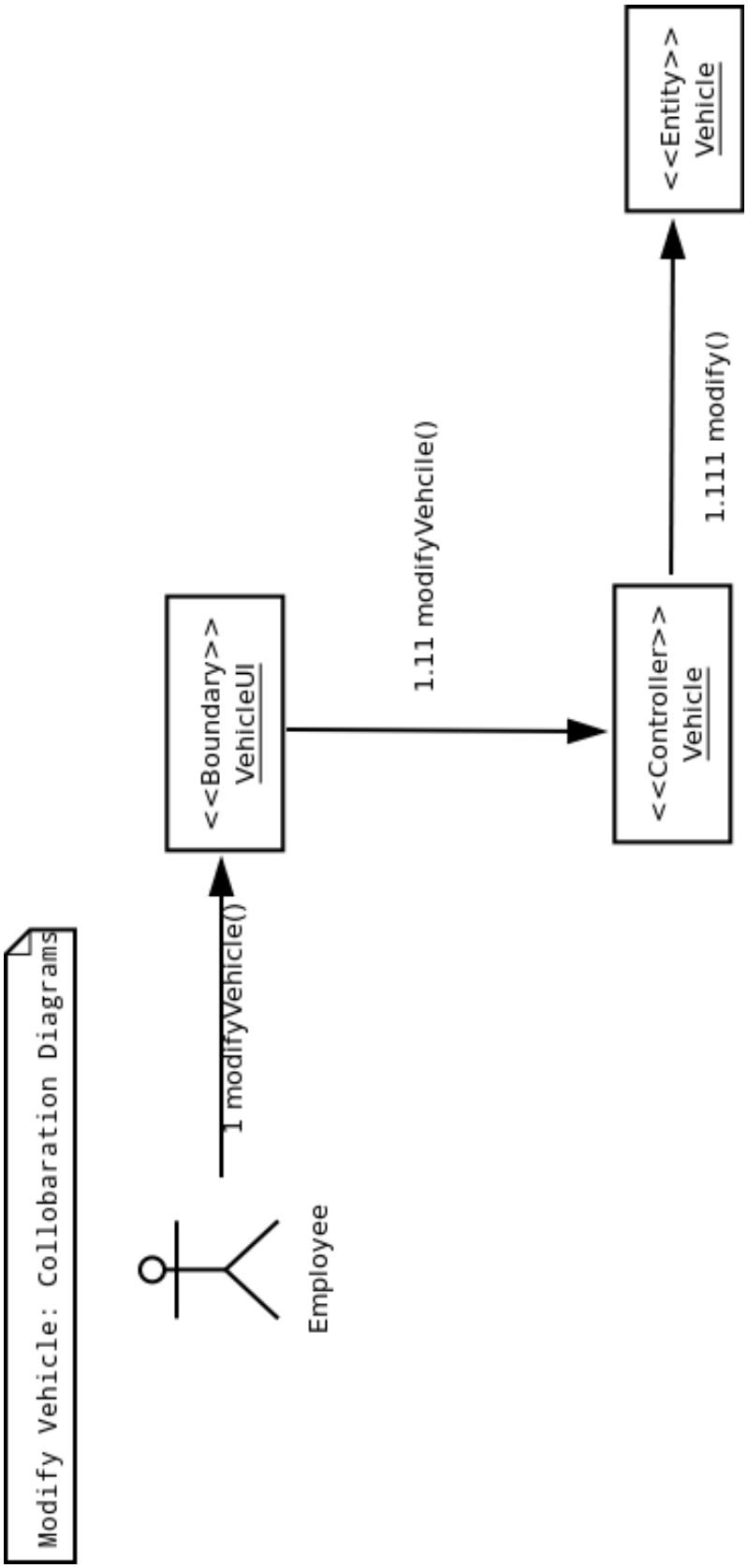
2.4.4.4 ListVehicle – Communication Diagram – Use Case: Page 27



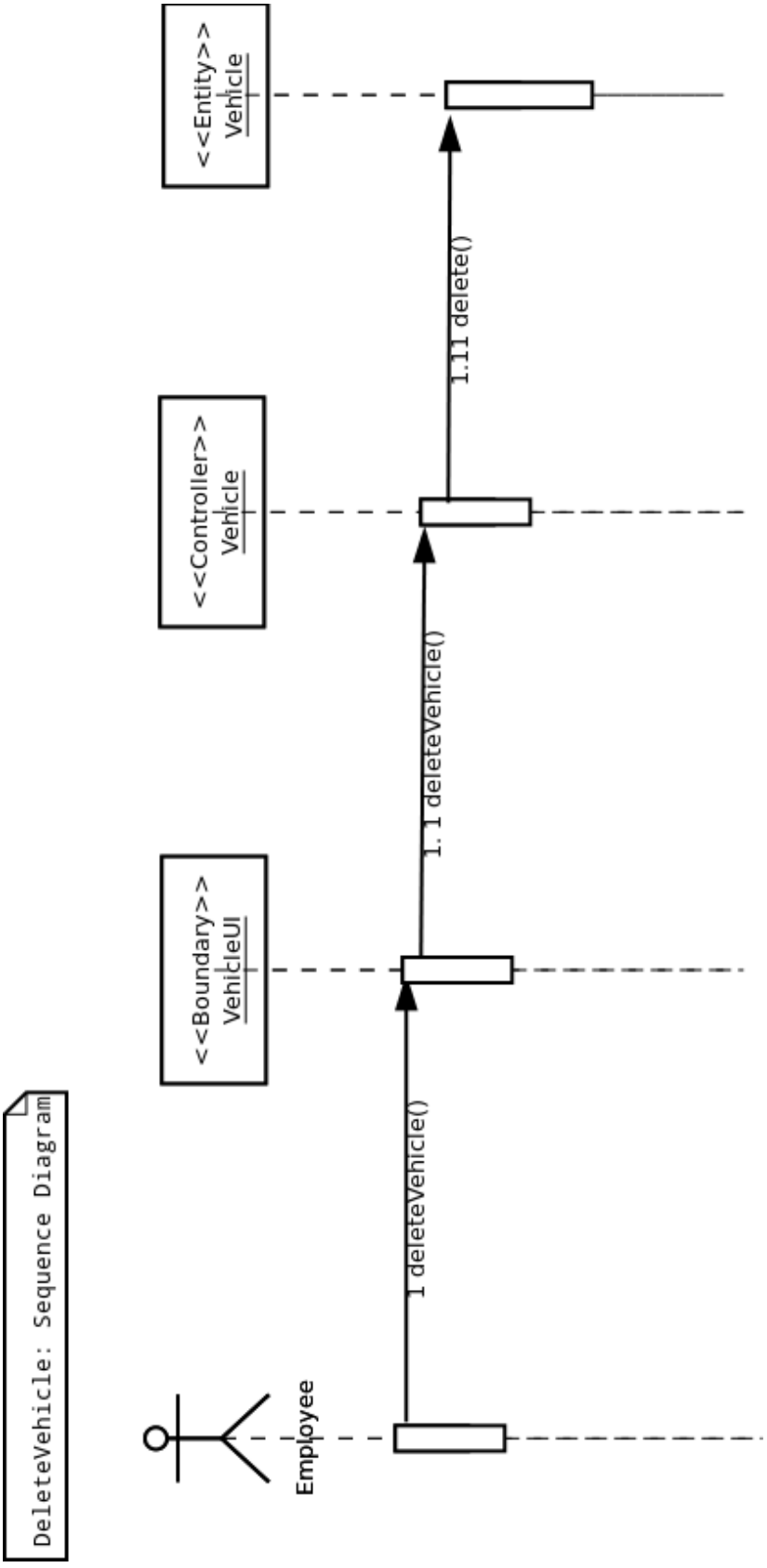
2.4.4.5 ModifyVehicle – Sequence Diagram – Use Case: Page 28



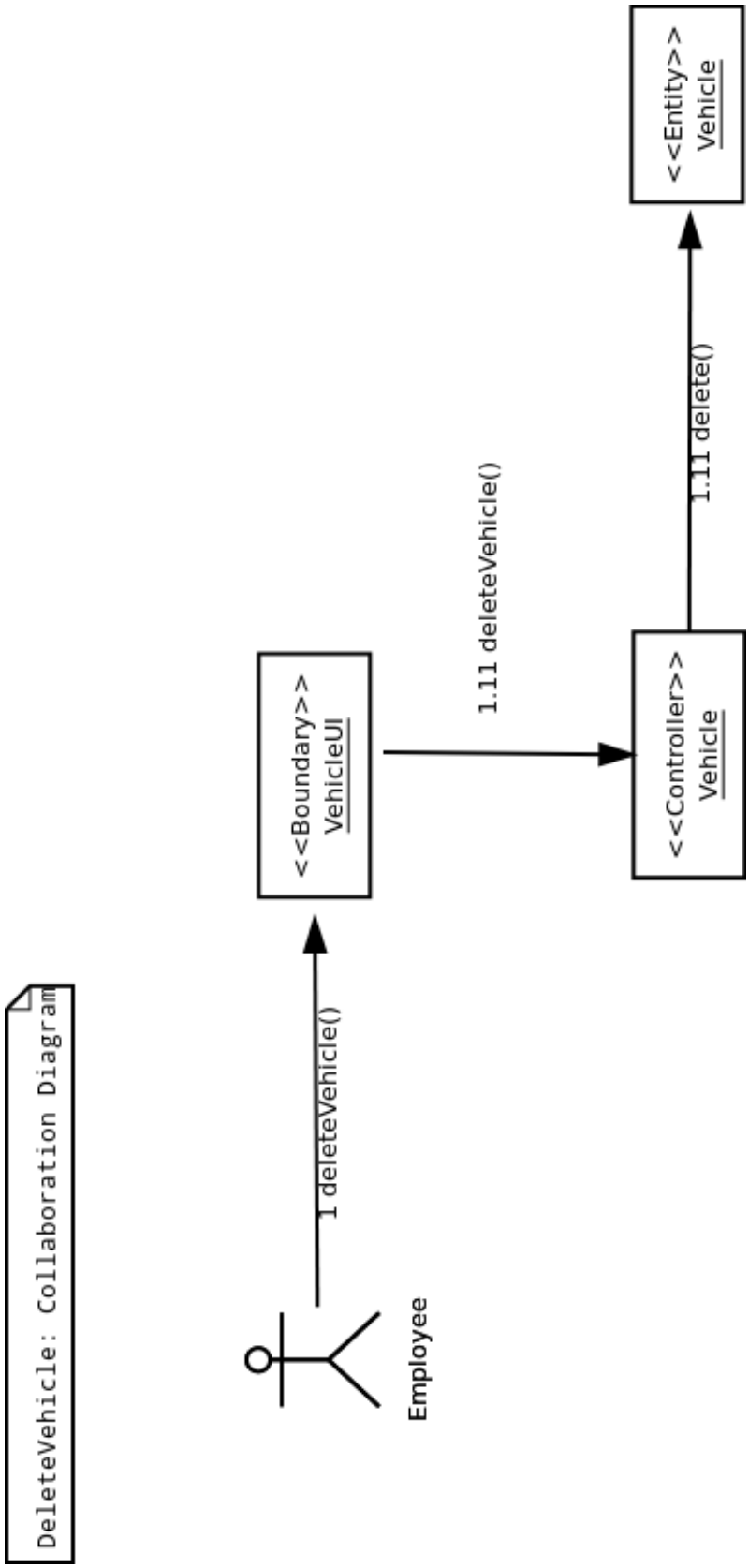
2.4.4.5 ModifyVehicle – Communication Diagram – Use Case: Page 28



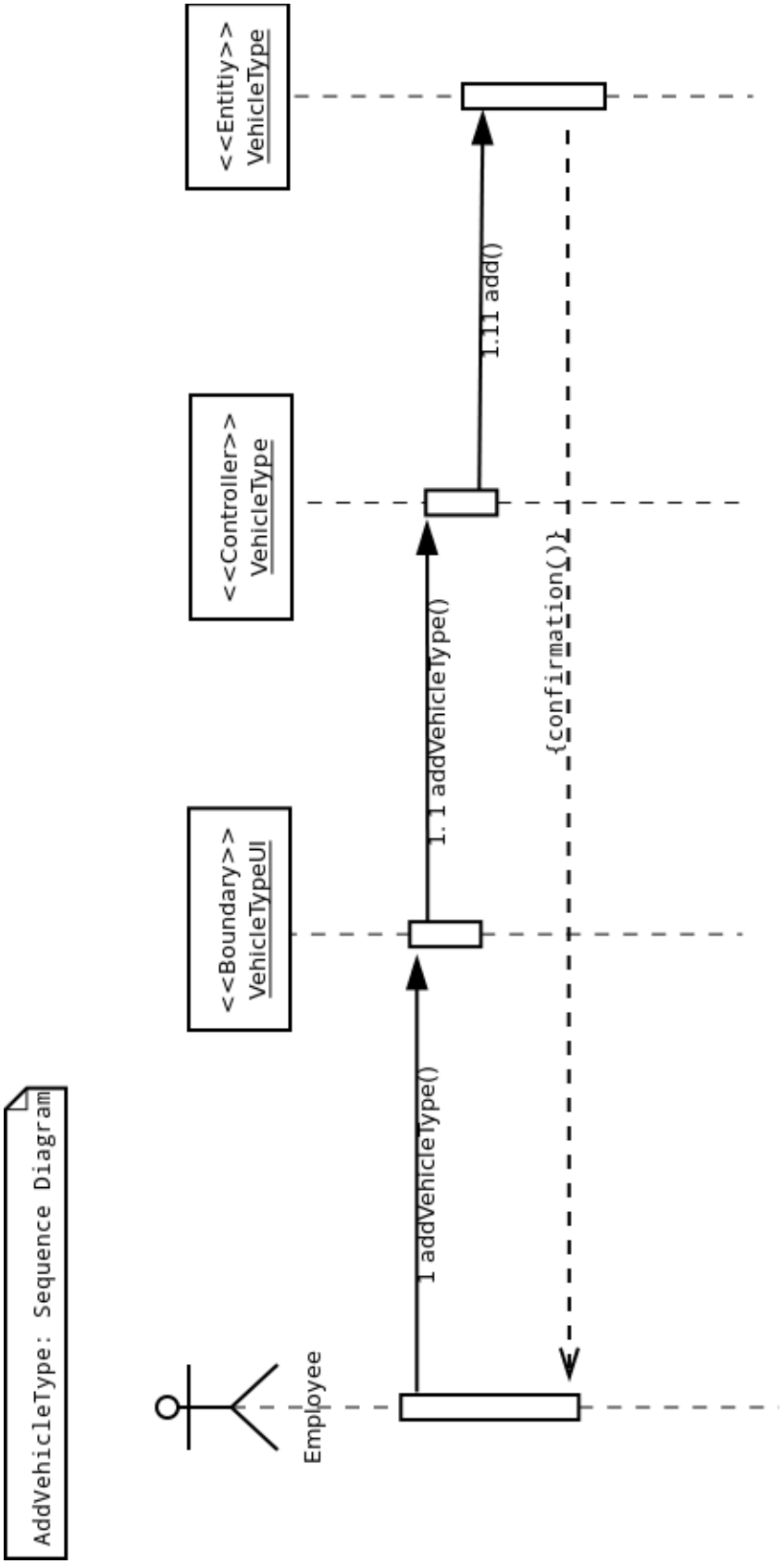
2.4.4.6 DeleteVehicle – Sequence Diagram – Use Case: Page 29



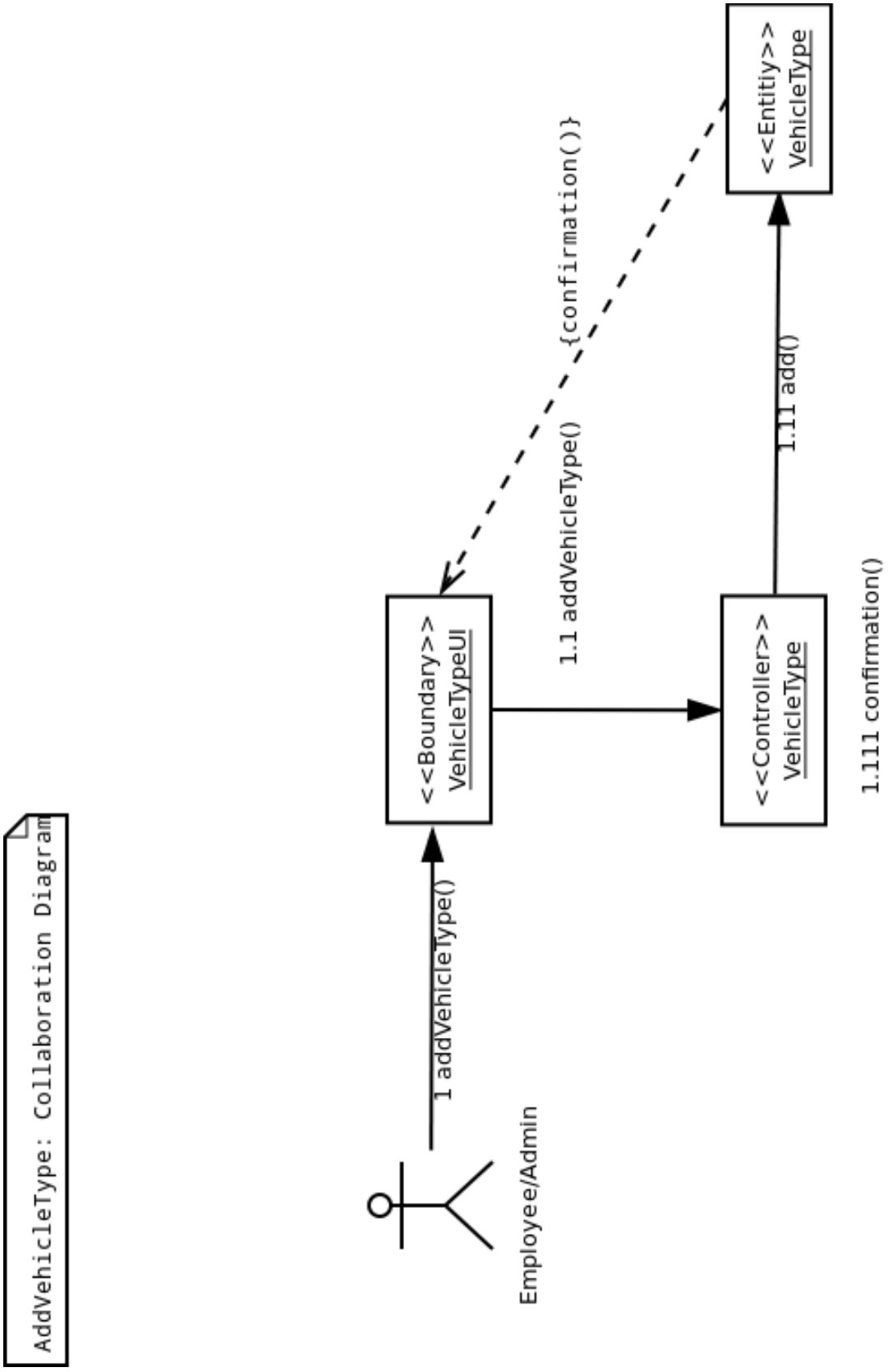
2.4.4.6 DeleteVehicle – Communication Diagram – Use Case: Page 29



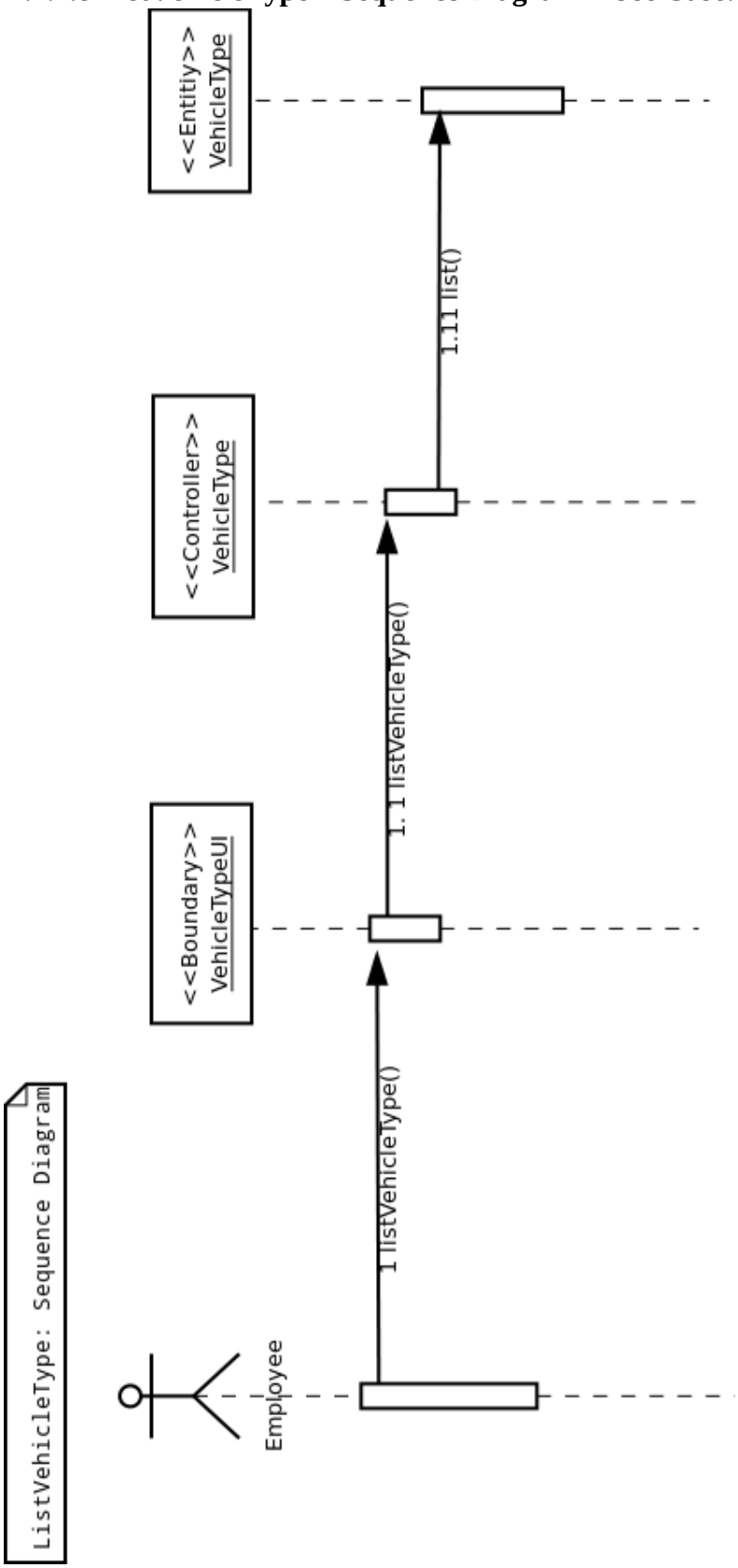
2.4.4.7 AddVehicleType – Sequence Diagram – Use Case: Page 30



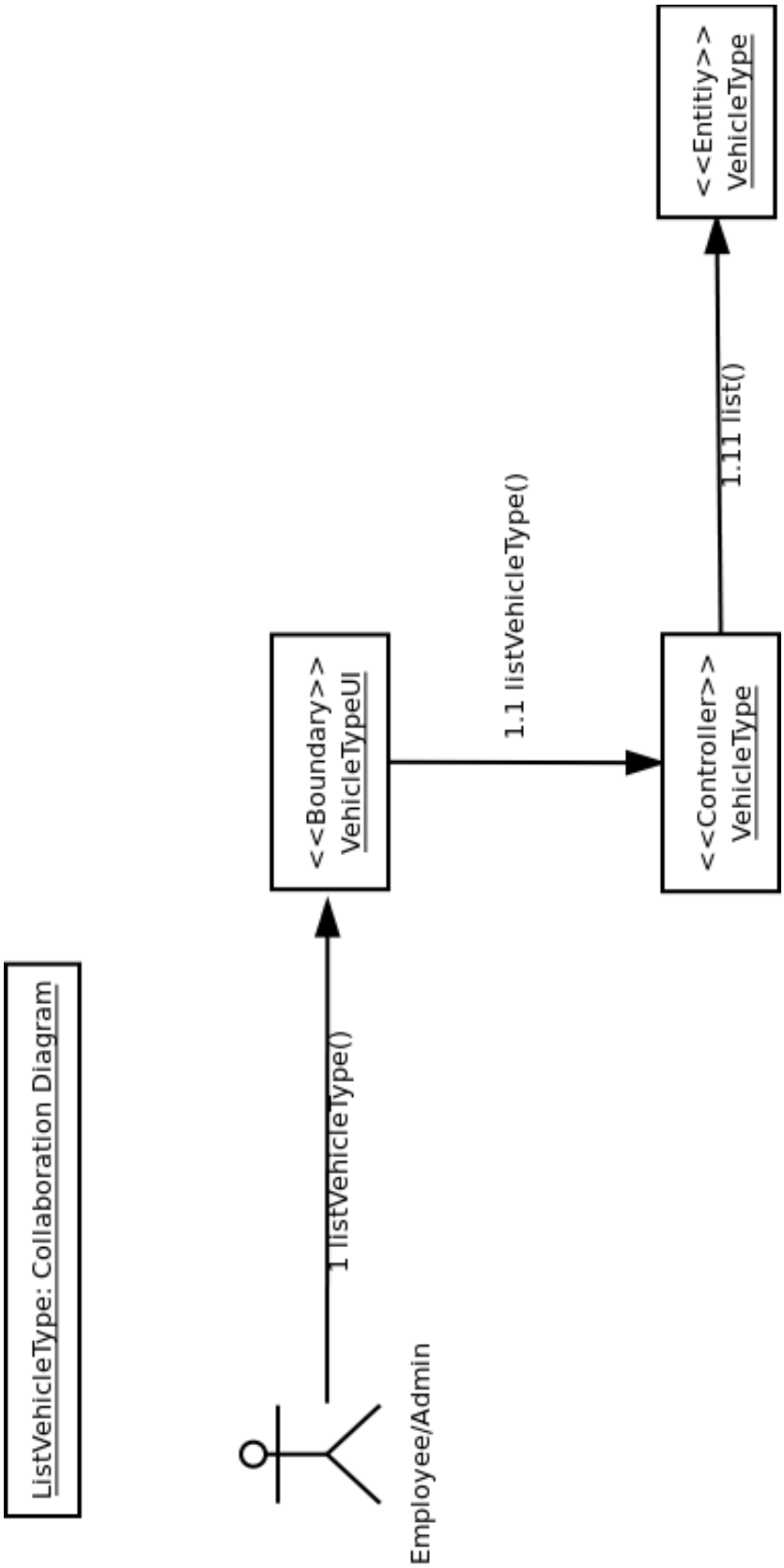
2.4.4.7 AddVehicleType – Communication Diagram – Use Case: Page 30



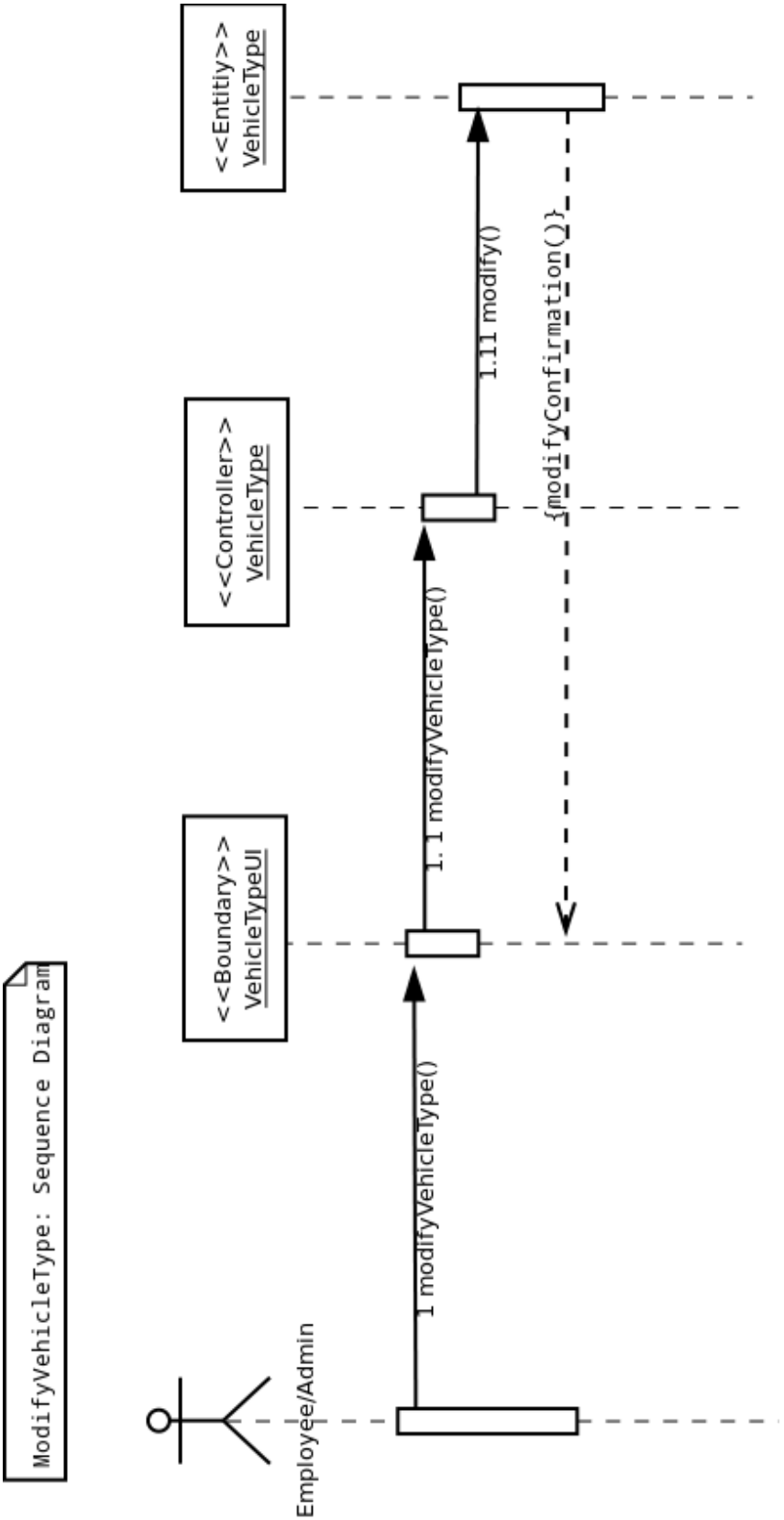
2.4.4.8 ListVehicleType – Sequence Diagram – Use Case: Page 32



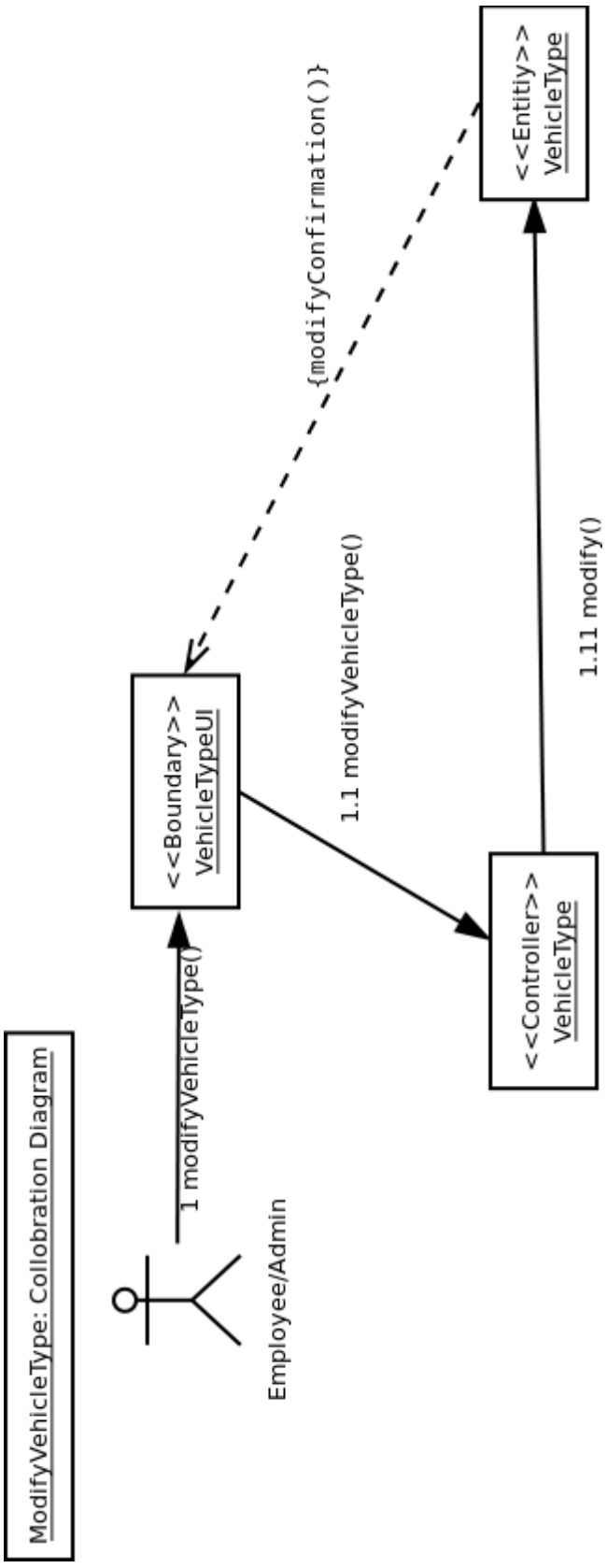
2.4.4.8 ListVehicleType – Communication Diagram – Use Case: Page 32



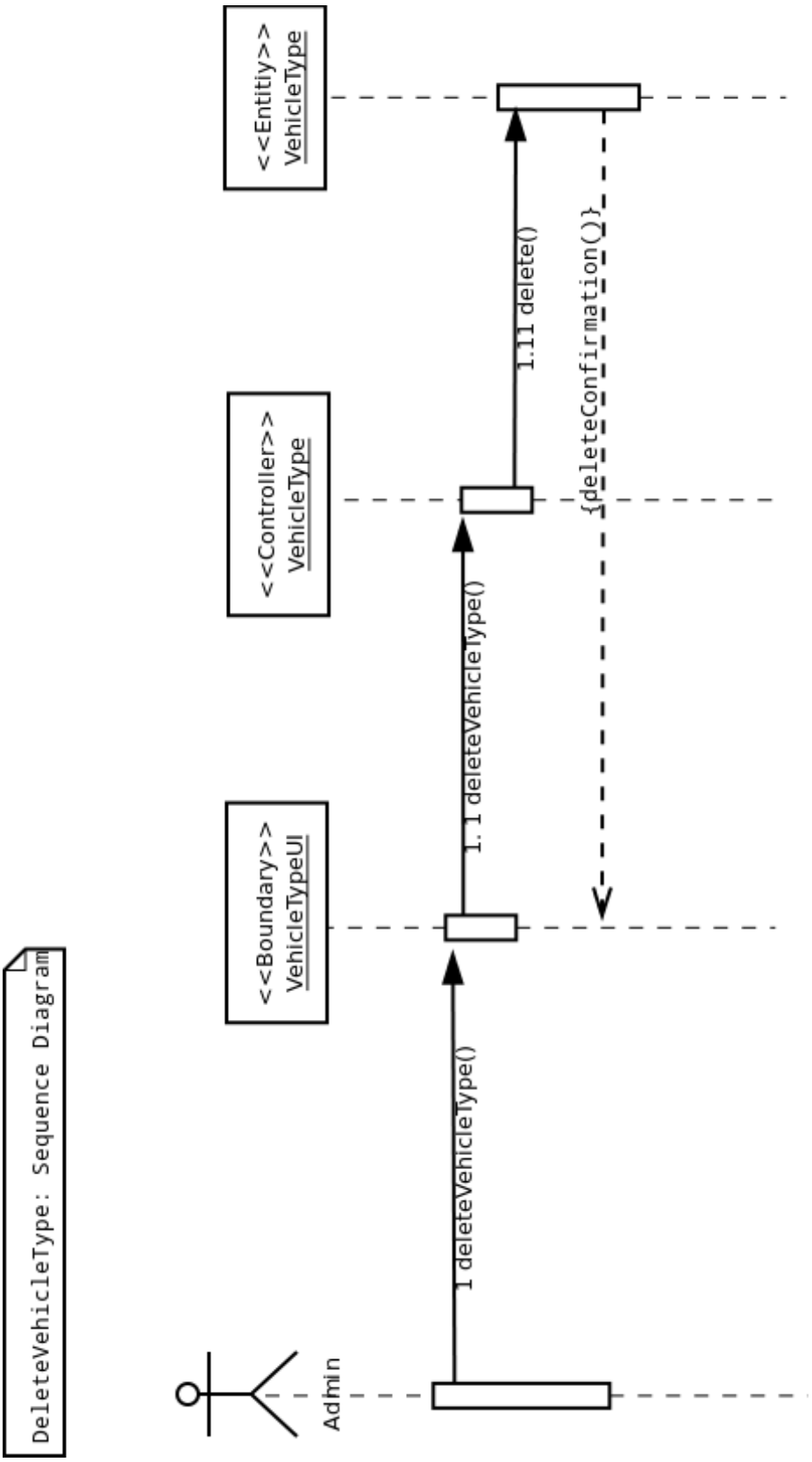
2.4.4.9 ModifyVehicleType – Sequence Diagram – Use Case: Page 33



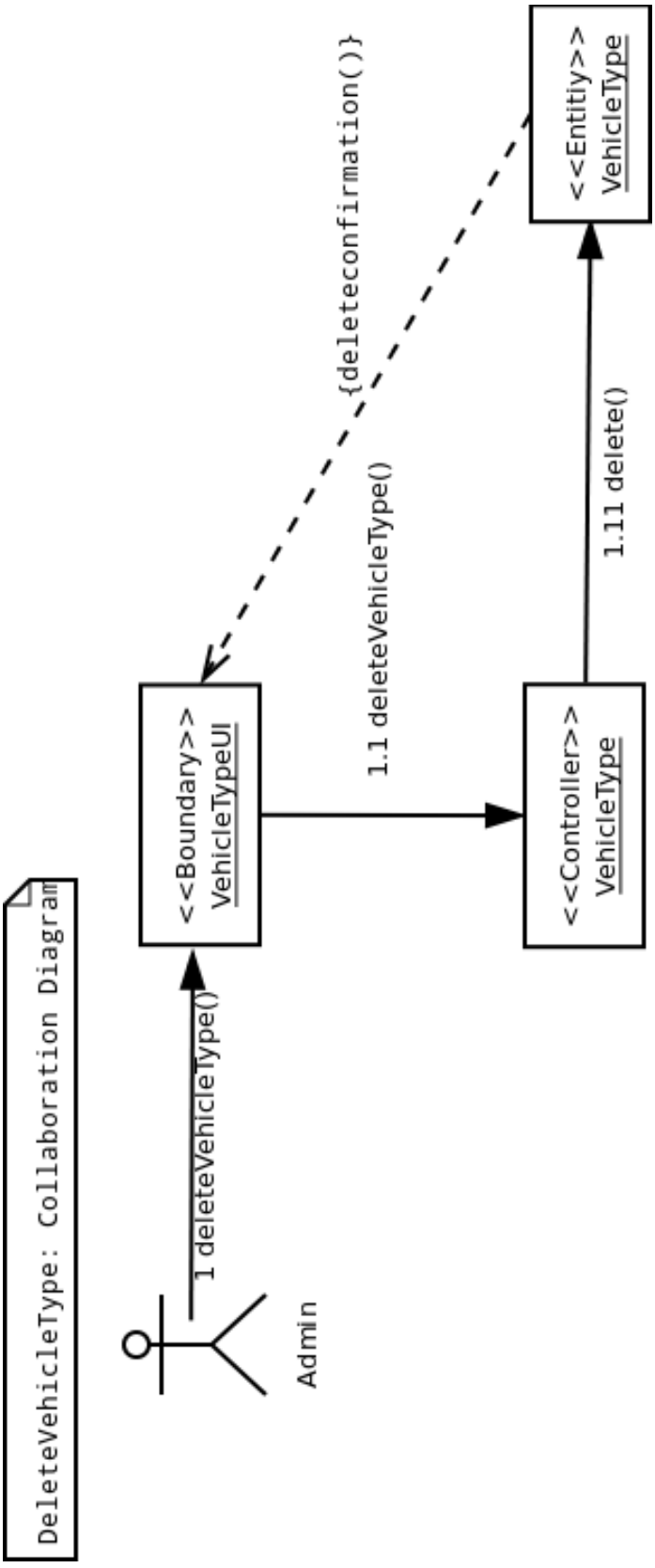
2.4.4.9 ModifyVehicleType – Communication Diagram – Use Case: Page 33



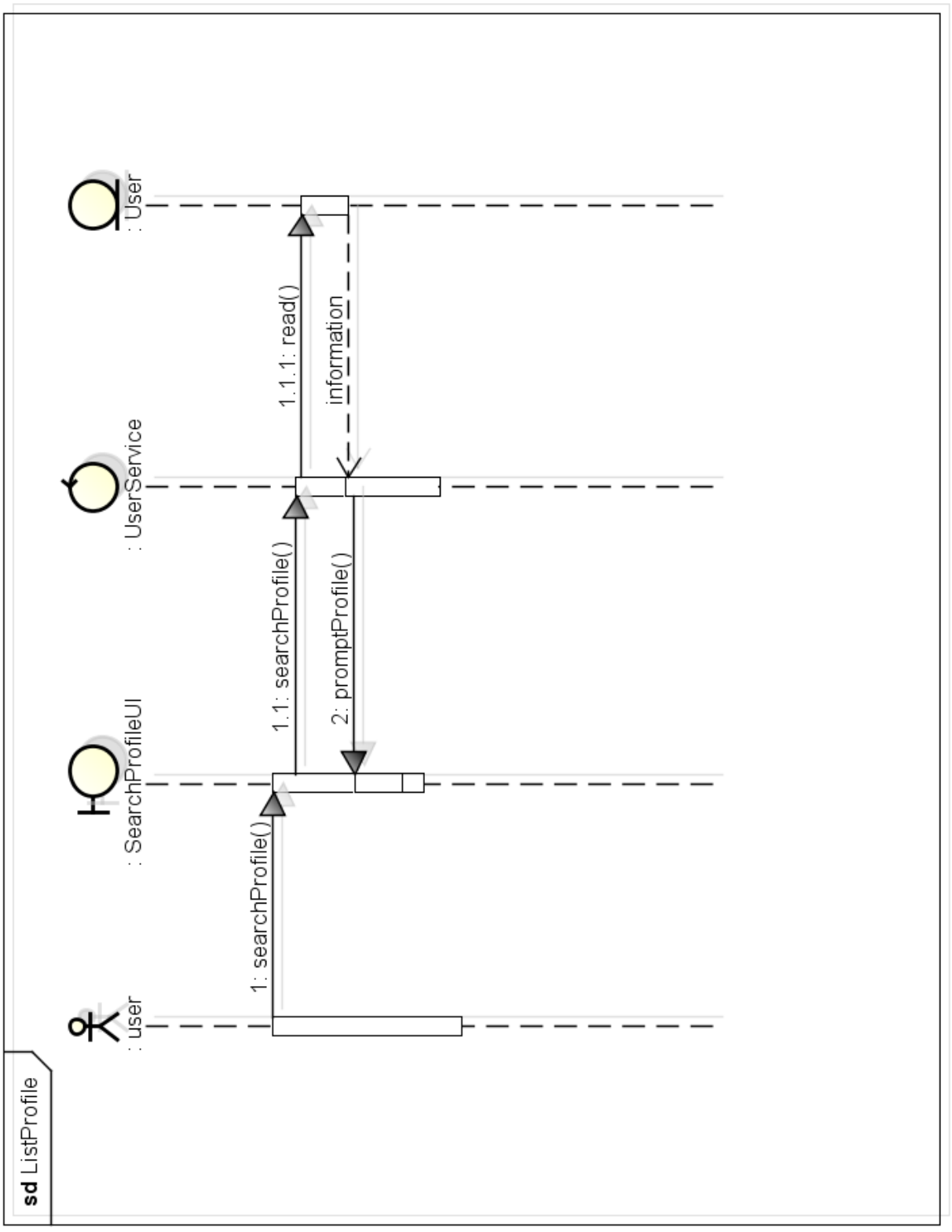
2.4.4.10 DeleteVehicleType – Sequence Diagram – Use Case: Page 34



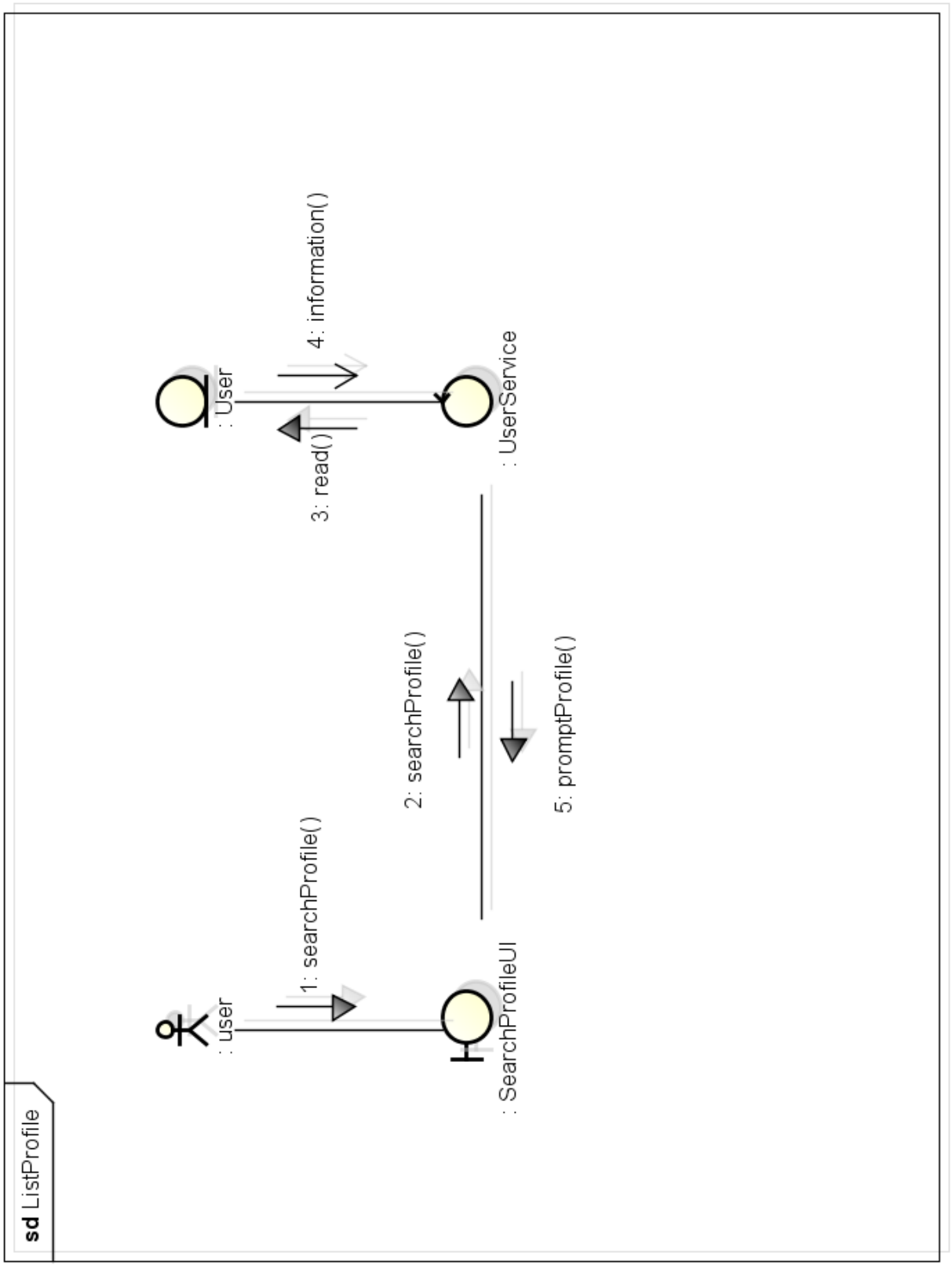
2.4.4.10 DeleteVehicleType – Communication Diagram – Use Case: Page 34



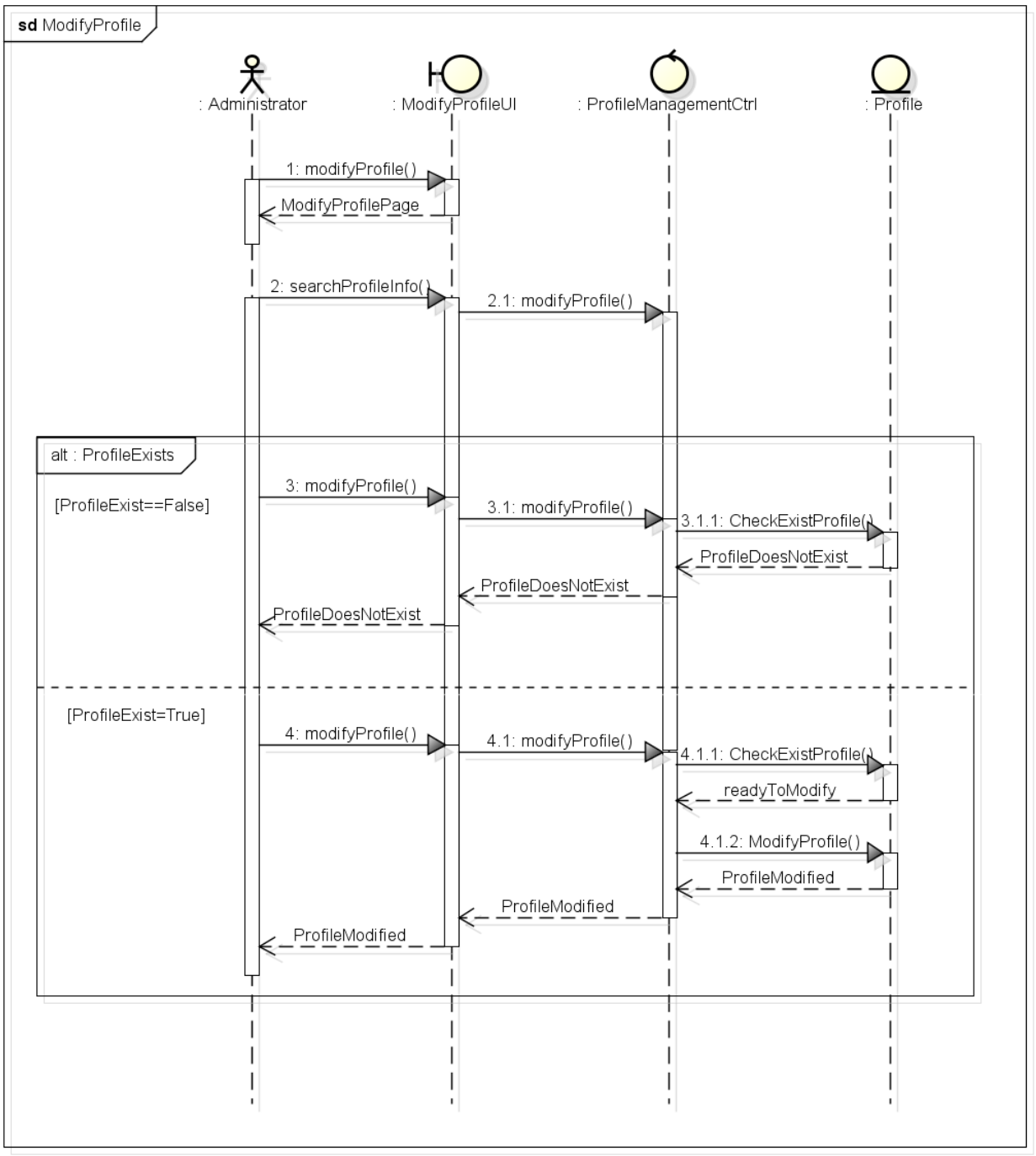
2.4.4.11 ListProfile – Sequence Diagram – Use Case: Page 35



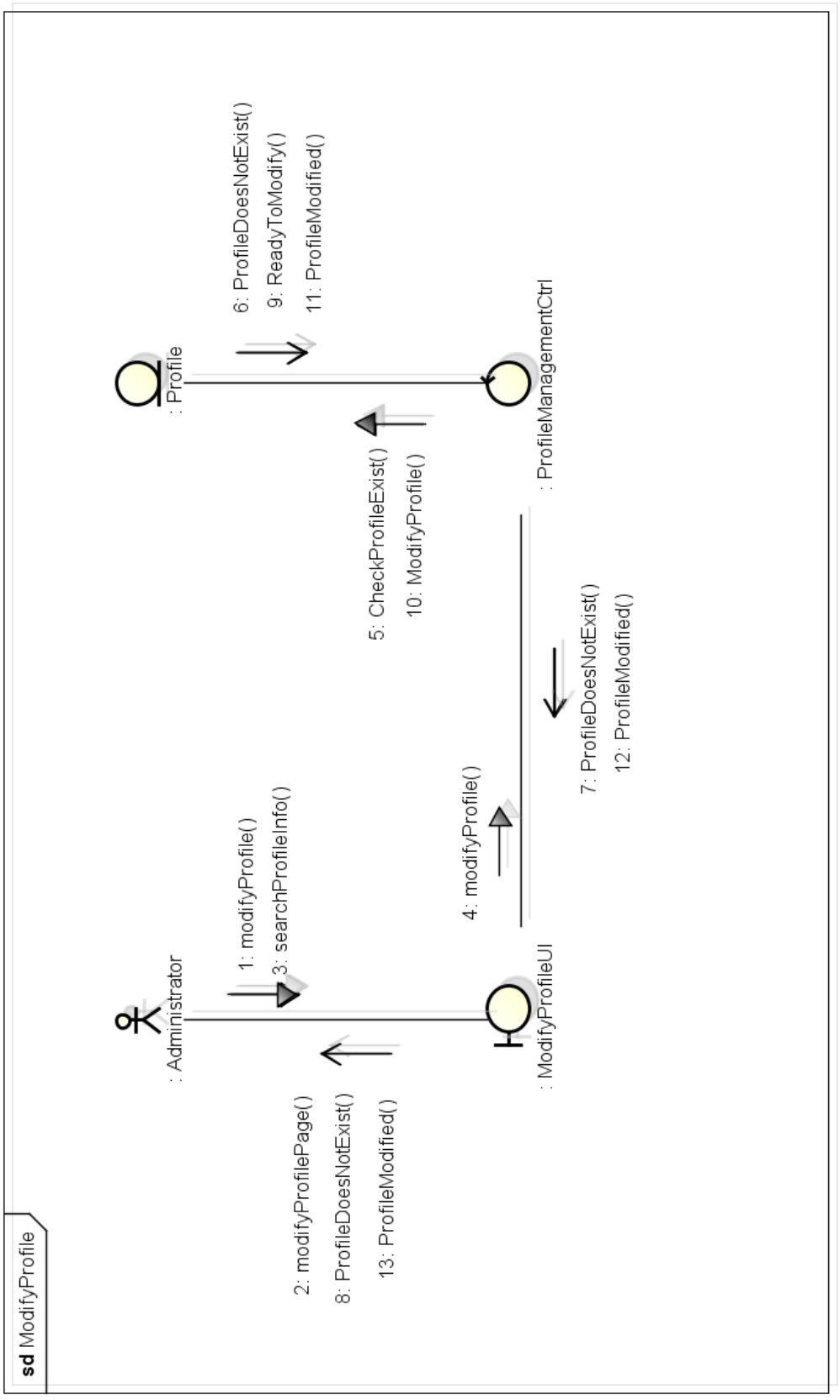
2.4.4.11 ListProfile – Communication Diagram – Use Case: Page 35



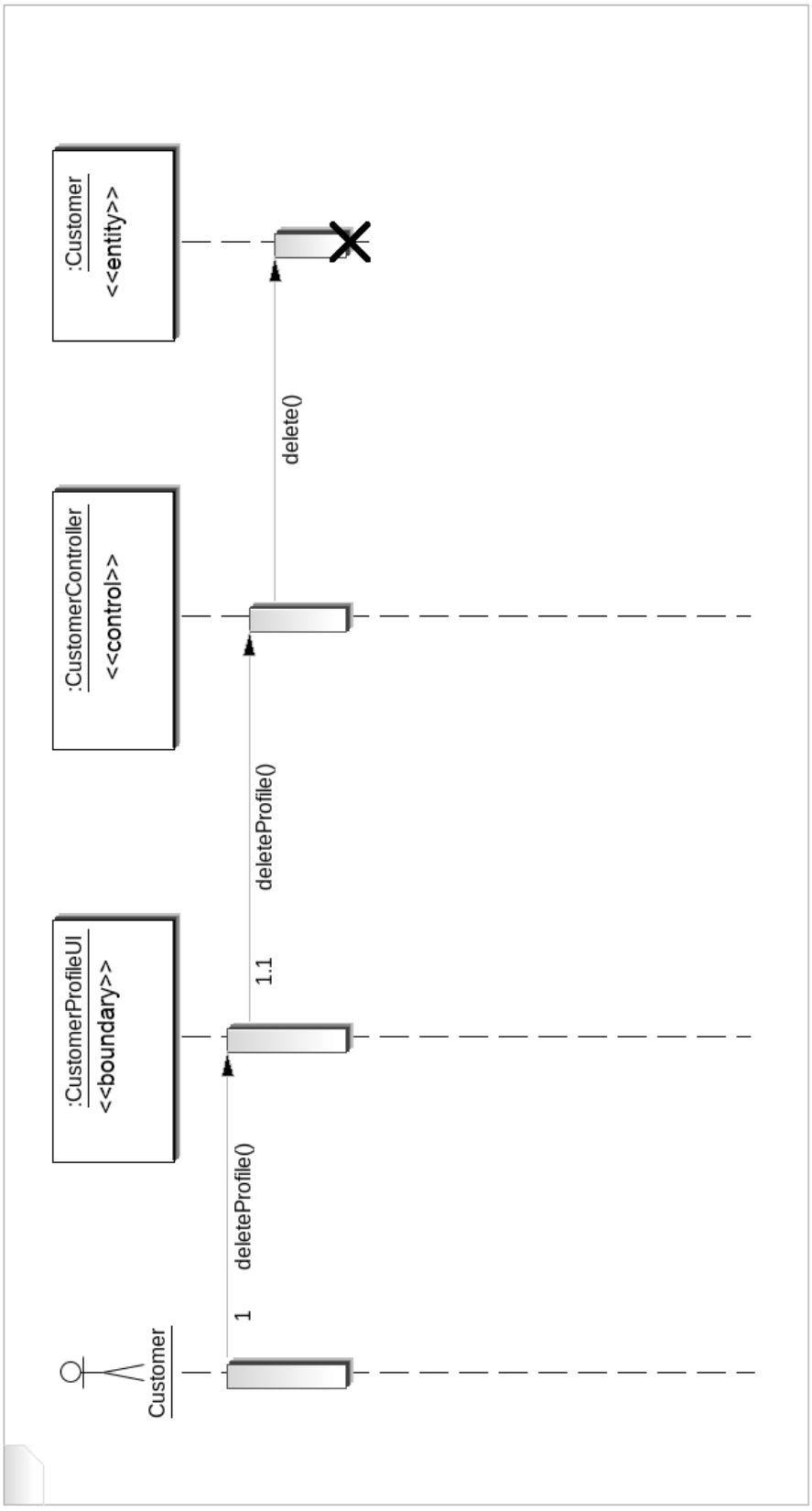
2.4.4.12 ModifyProfile – Sequence Diagram – Use Case: Page 36



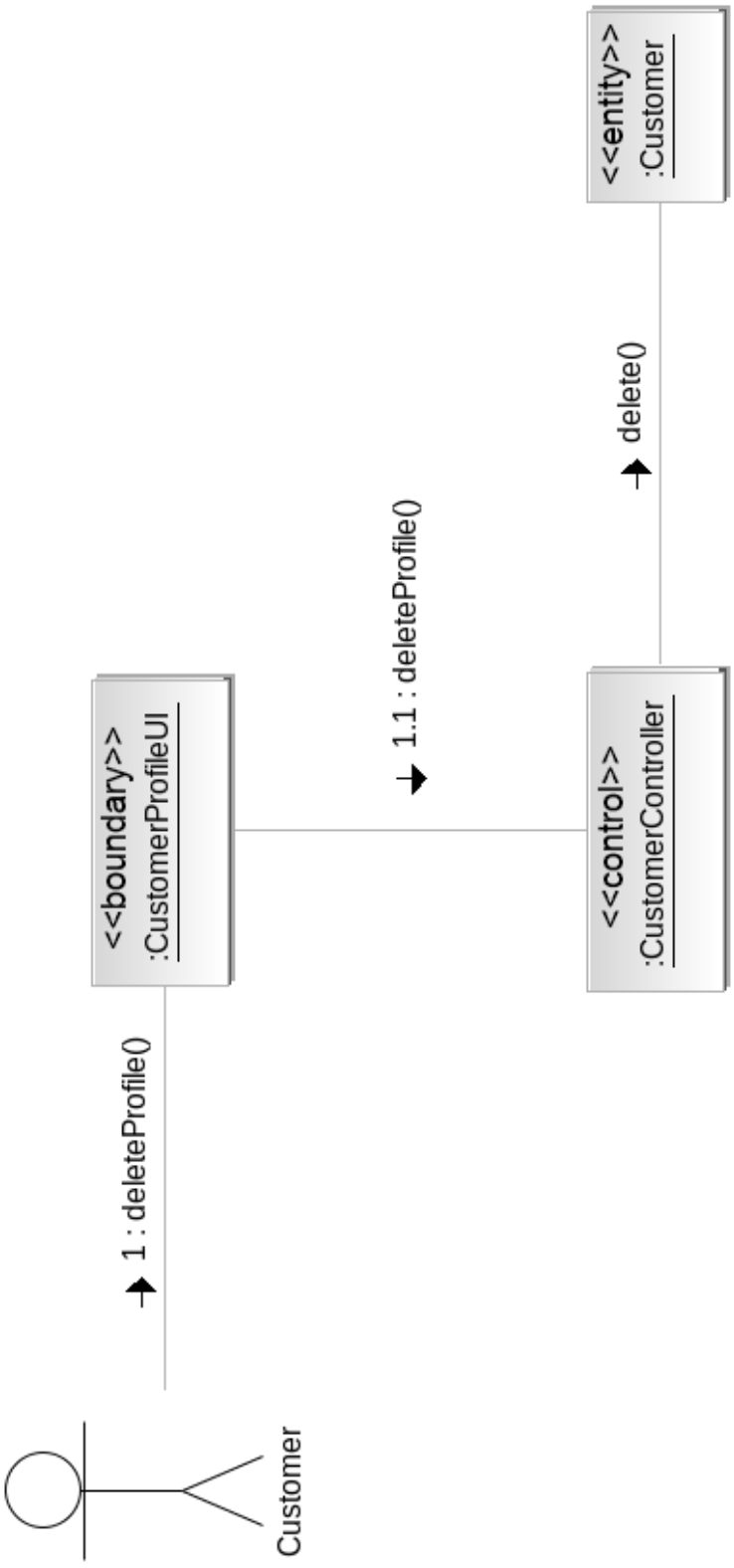
2.4.4.12 ModifyProfile – Communication Diagram – Use Case: Page 36



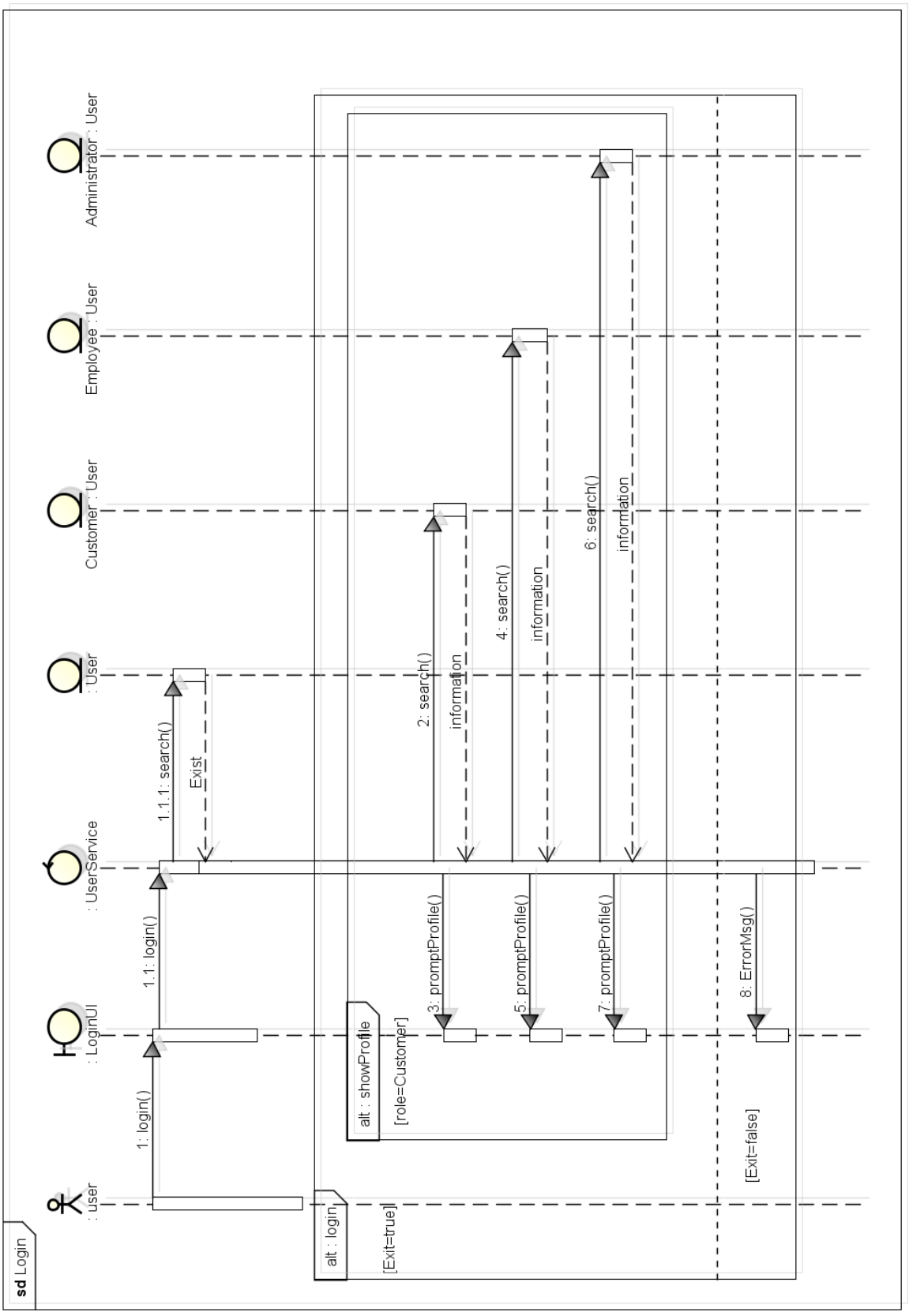
2.4.4.13 DeleteProfile – Sequence Diagram – Use Case: Page 37



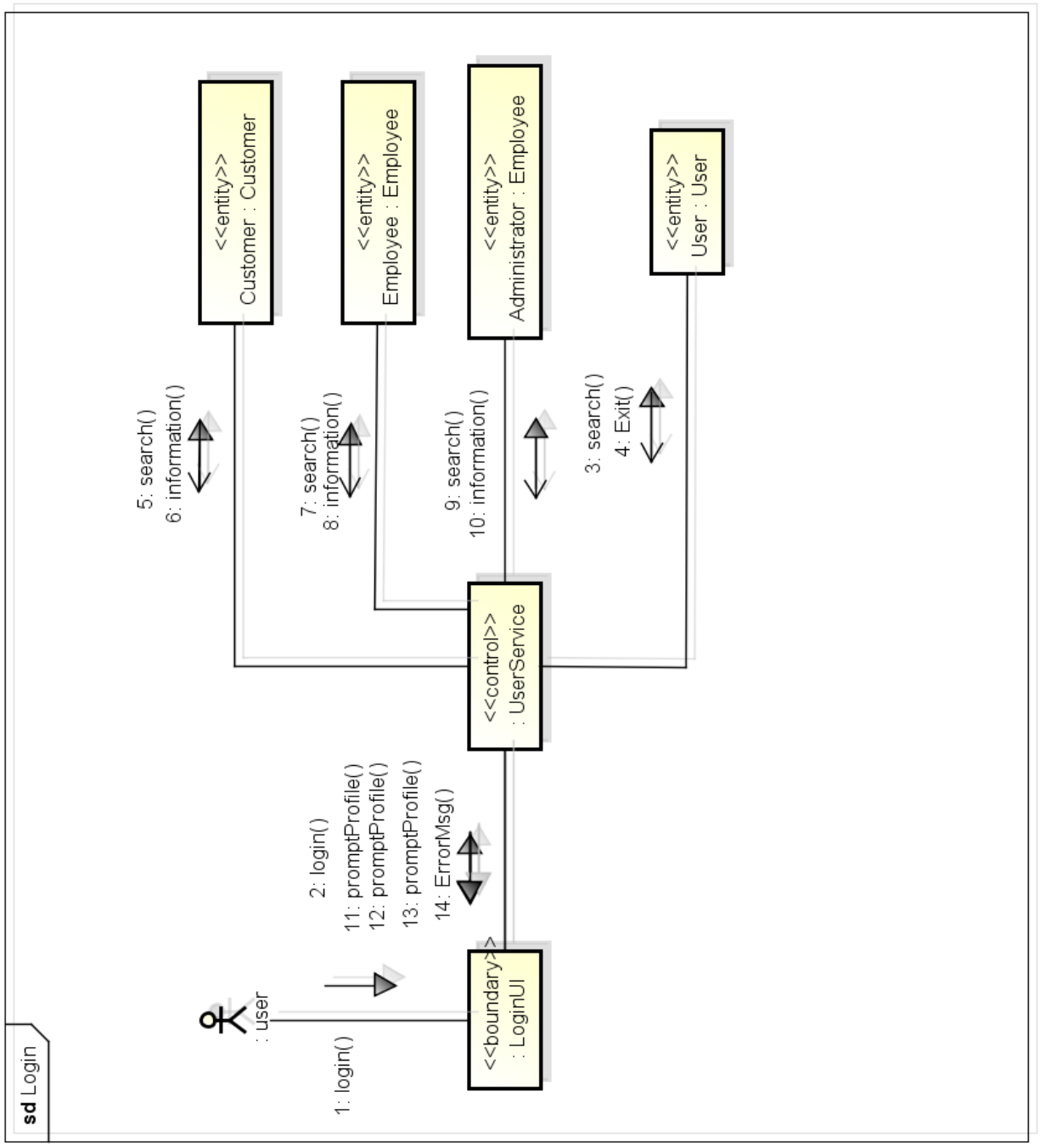
2.4.4.13 DeleteProfile – Communication Diagram – Use Case: Page 37



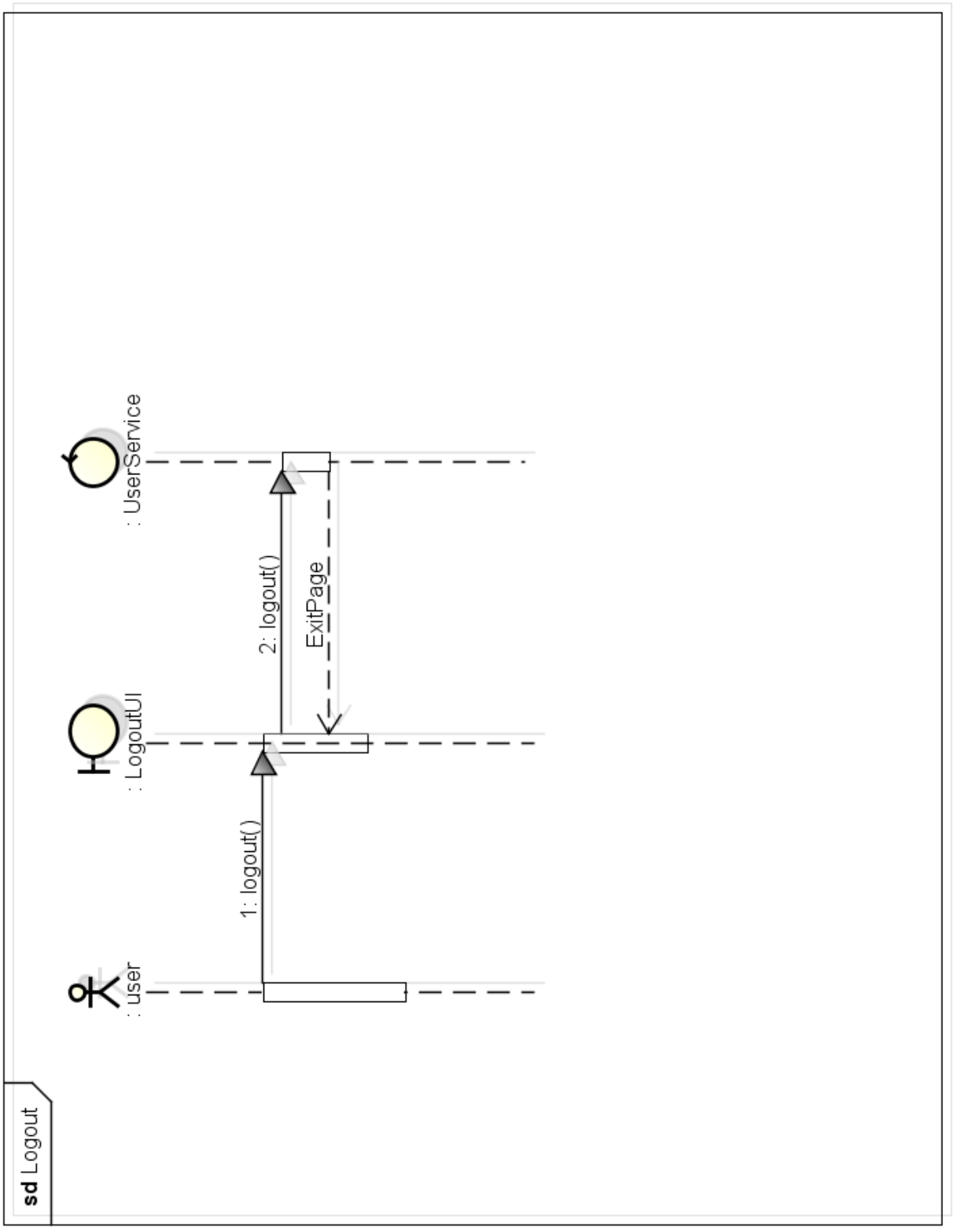
2.4.4.14 Login – Sequence Diagram – Use Case: Page 38



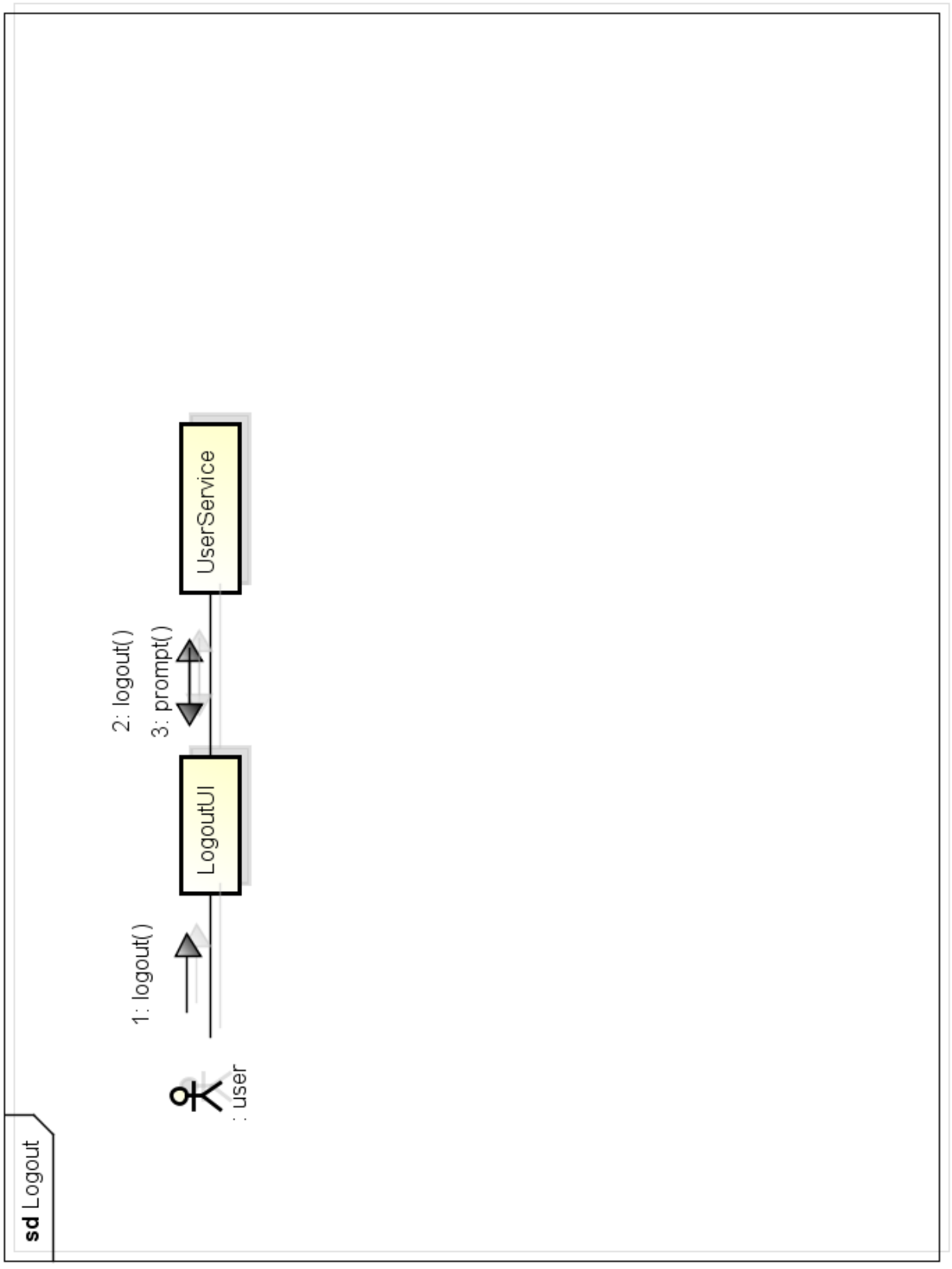
2.4.4.14 Login – Communication Diagram – Use Case: Page 38



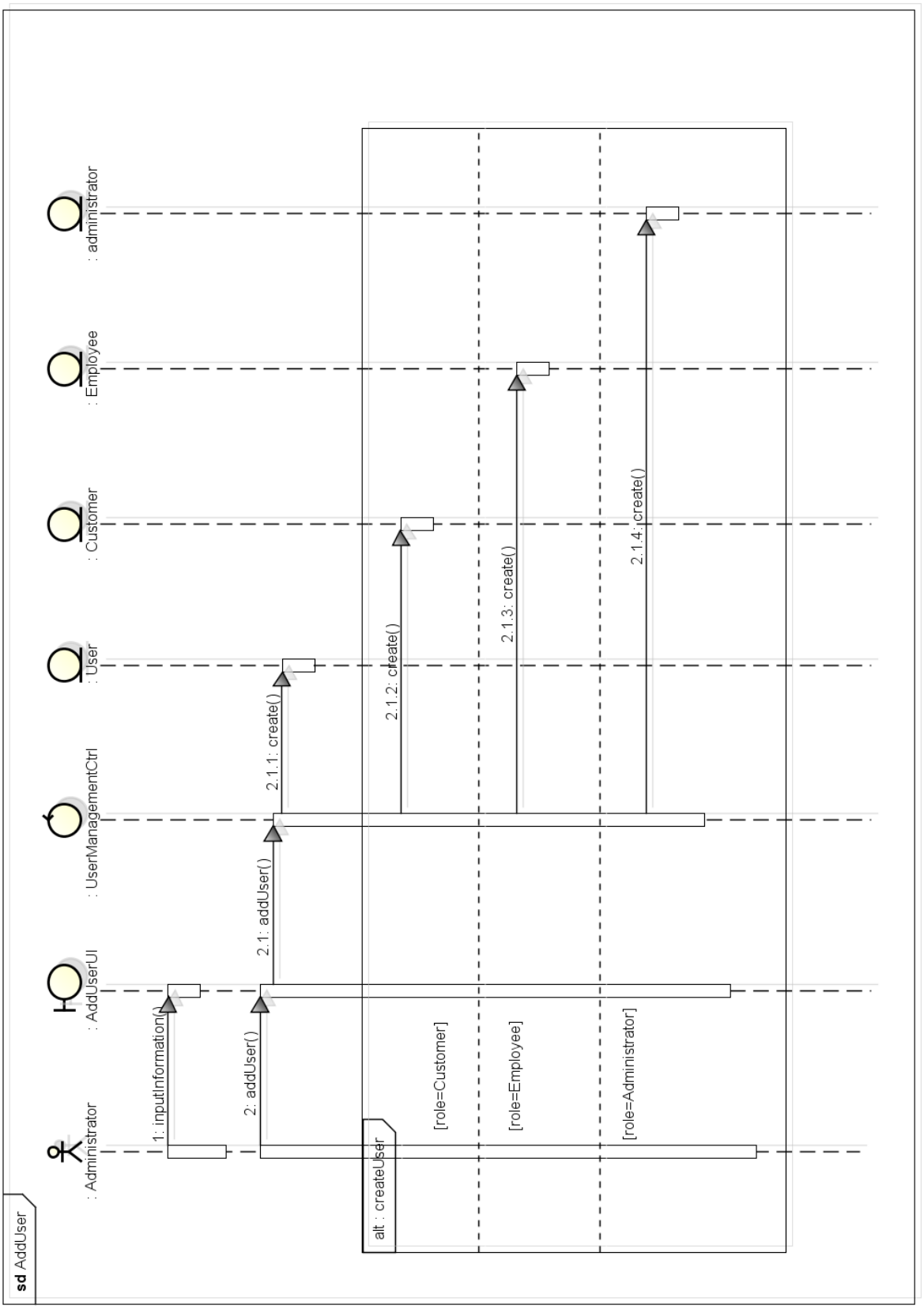
2.4.4.15 LogOut – Sequence Diagram – Use Case: Page 39



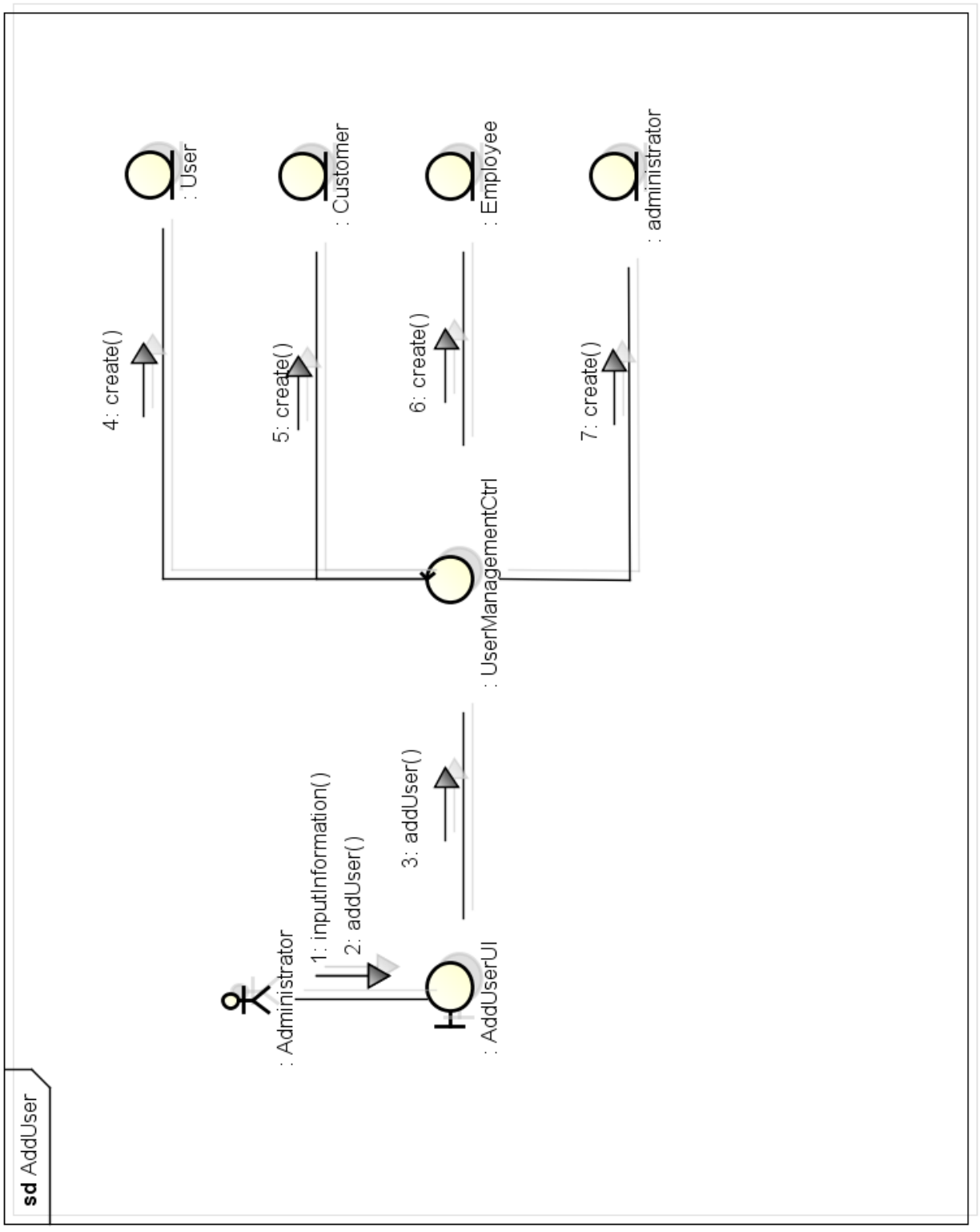
2.4.4.15 LogOut – Communication Diagram – Use Case: Page 39



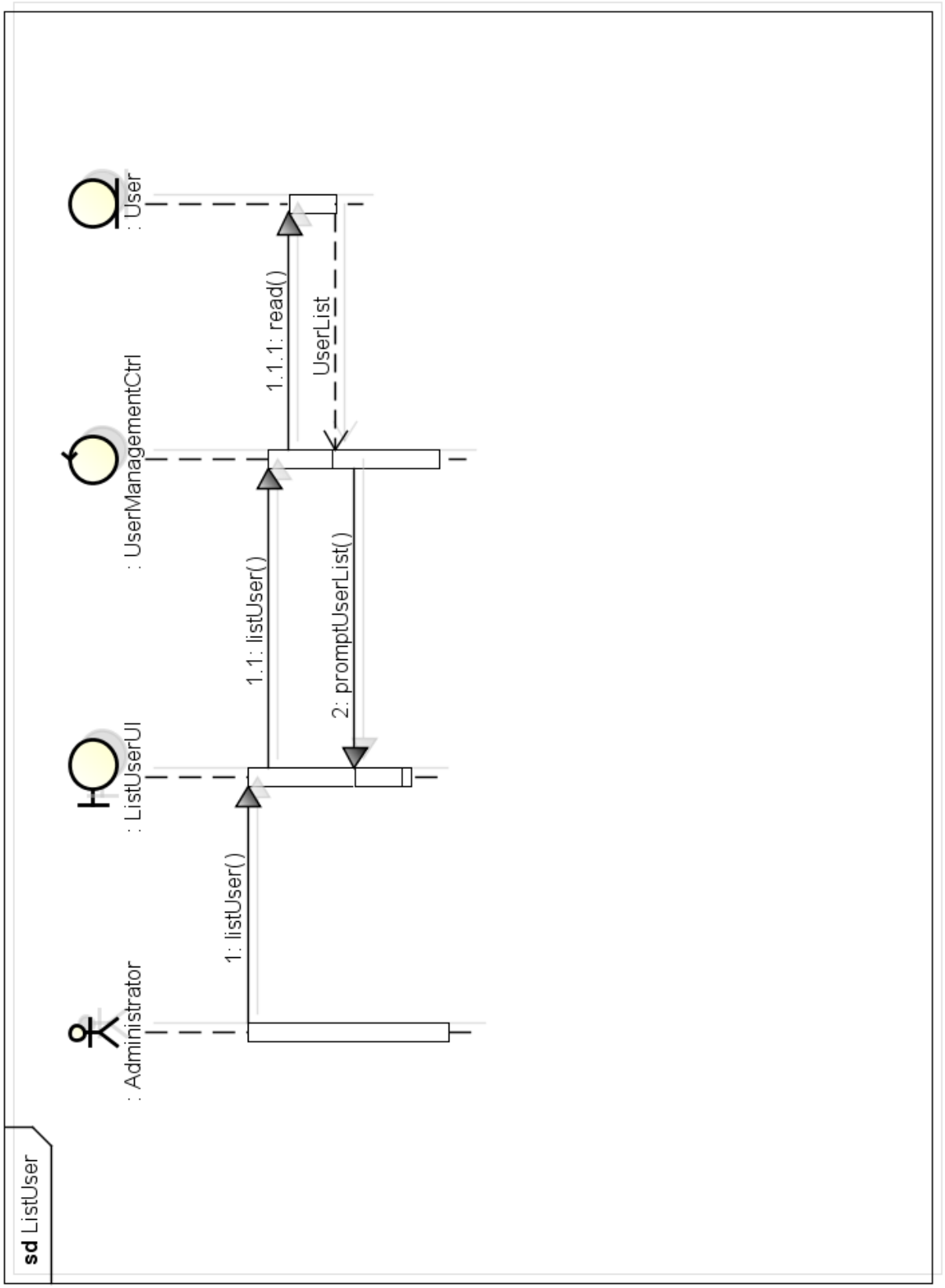
2.4.4.16 AddUser – Sequence Diagram – Use Case: Page 40



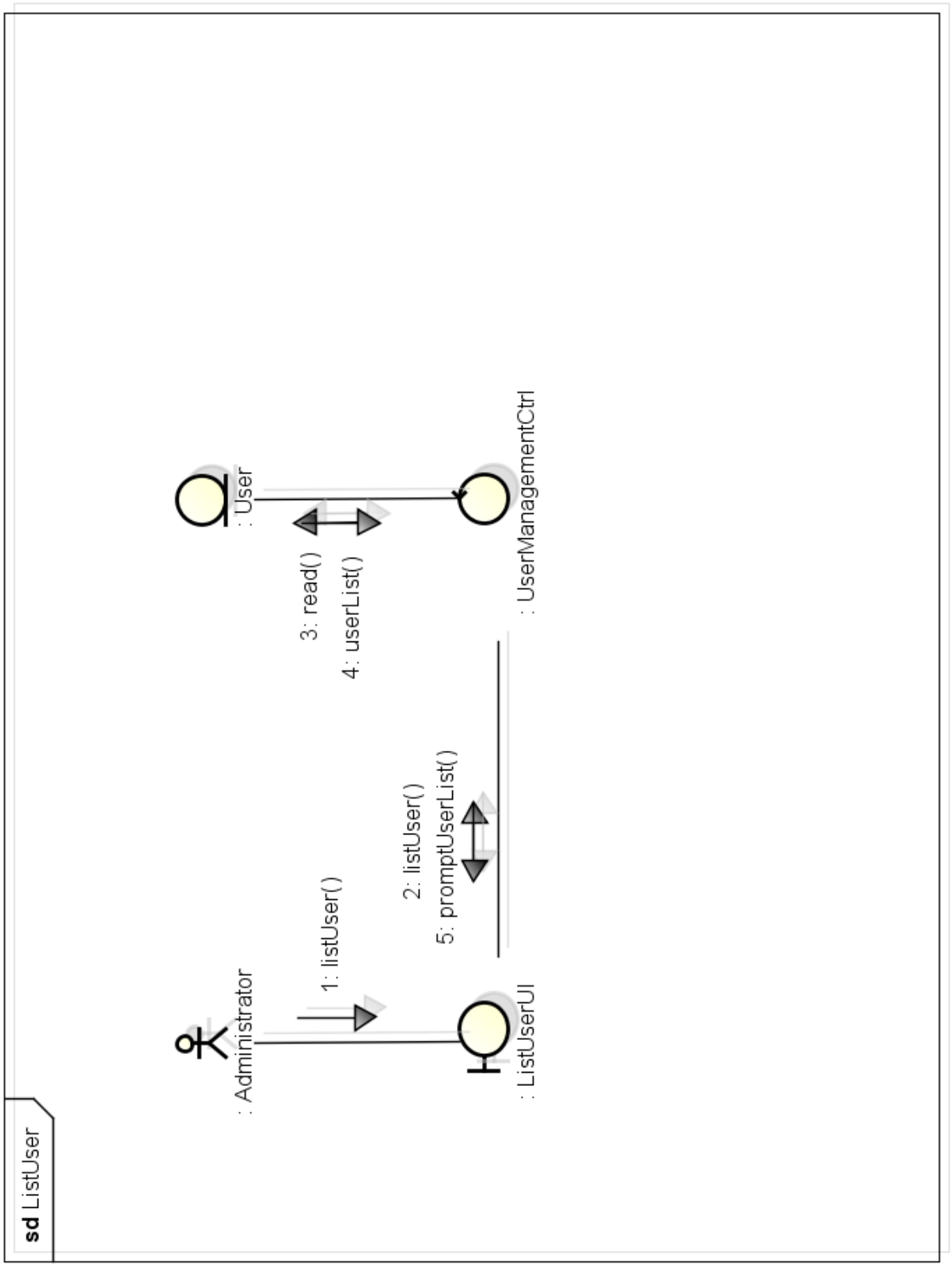
2.4.4.16 AddUser – Communication Diagram – Use Case: Page 40



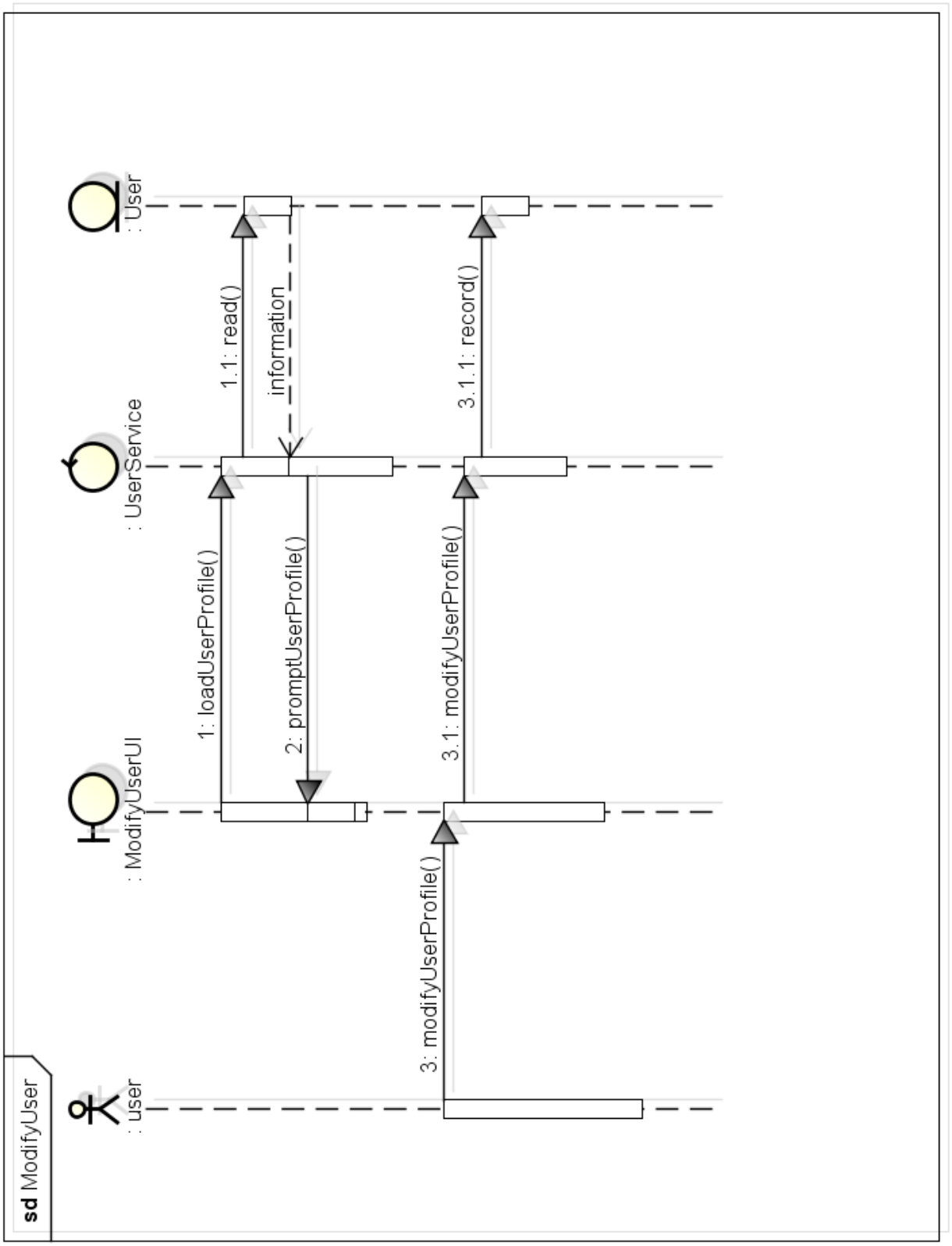
2.4.4.17 ListUser – Sequence Diagram – Use Case: Page 41



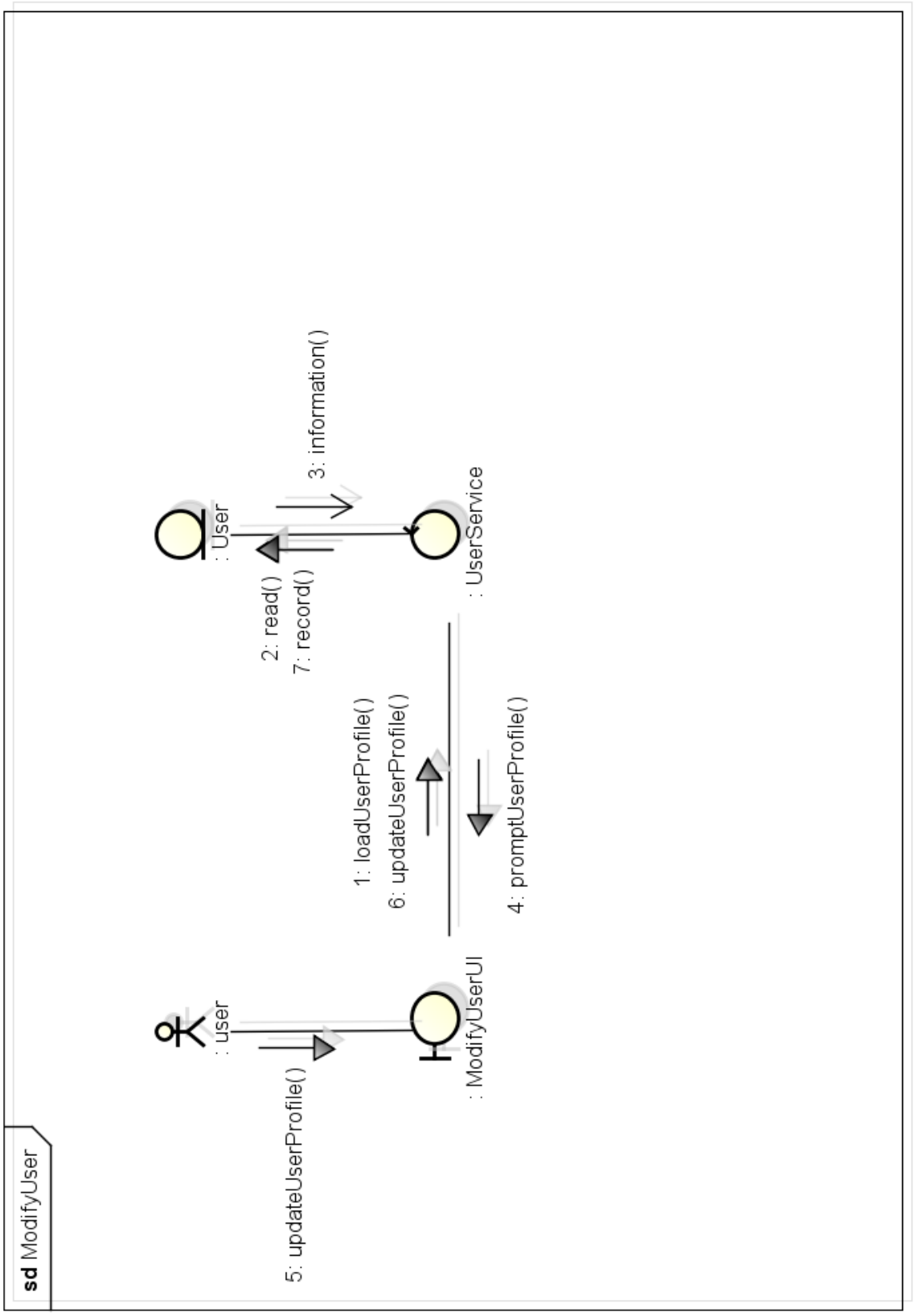
2.4.4.17 ListUser – Communication Diagram – Use Case: Page 41



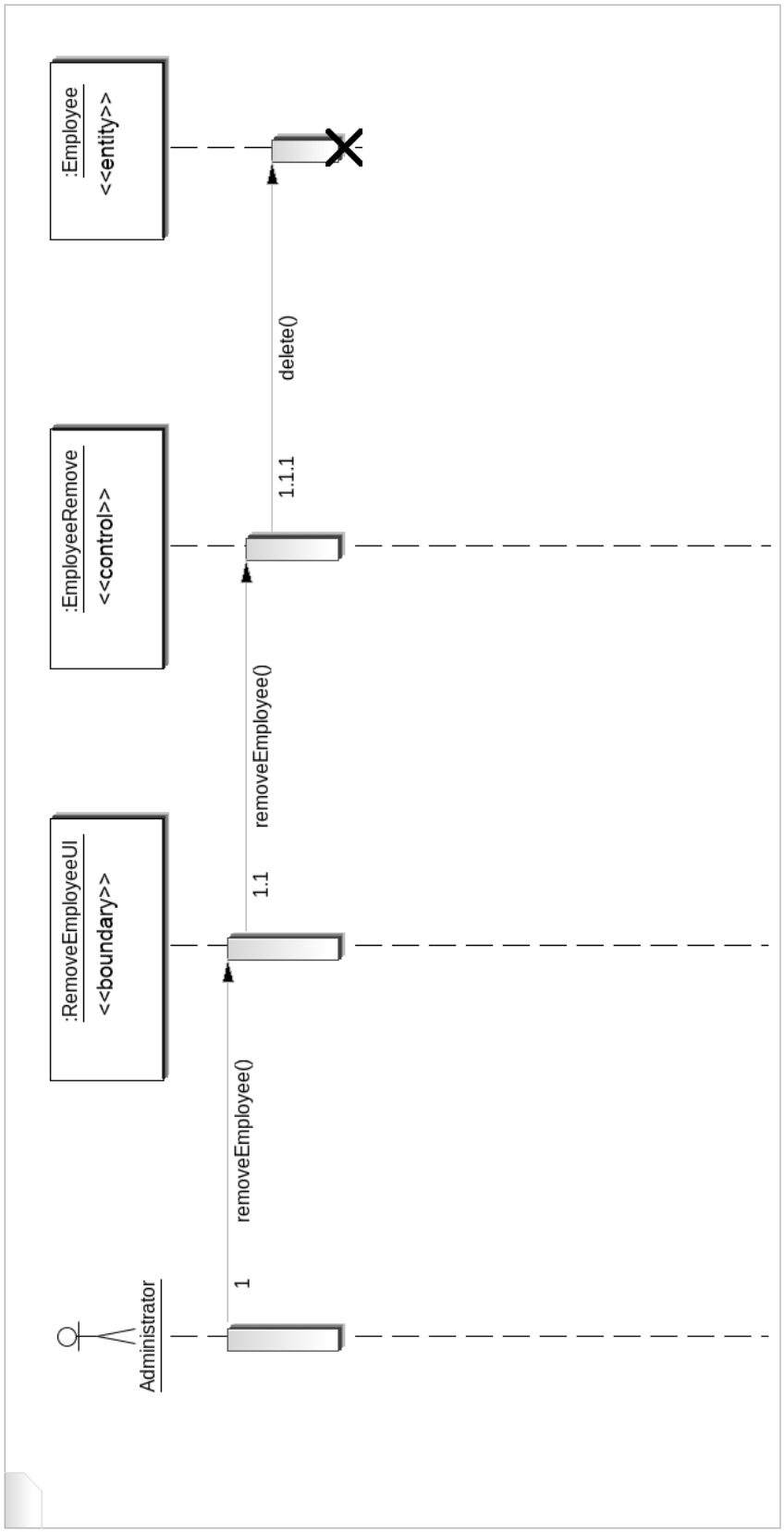
2.4.4.18 ModifyUser – Sequence Diagram – Use Case: Page 42



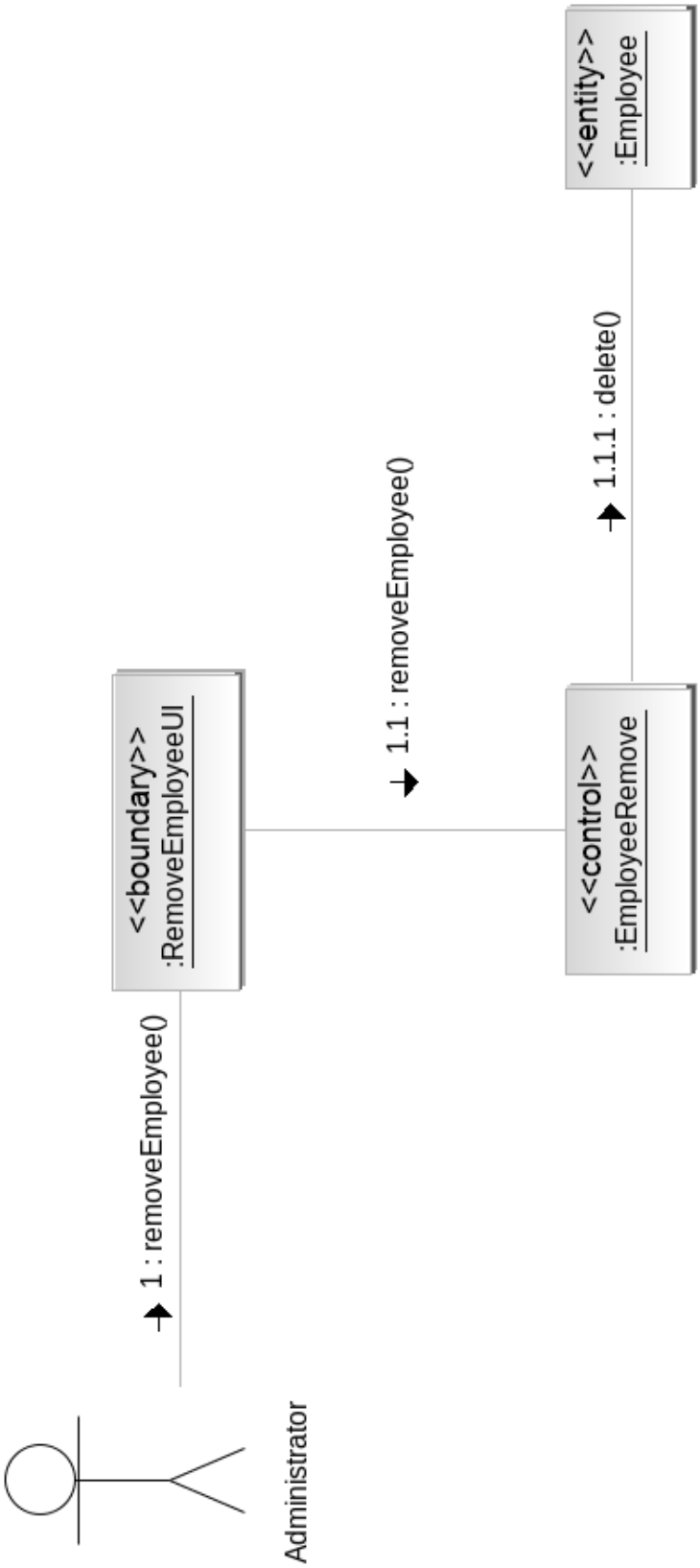
2.4.4.18 ModifyUser – Communication Diagram – Use Case: Page 42



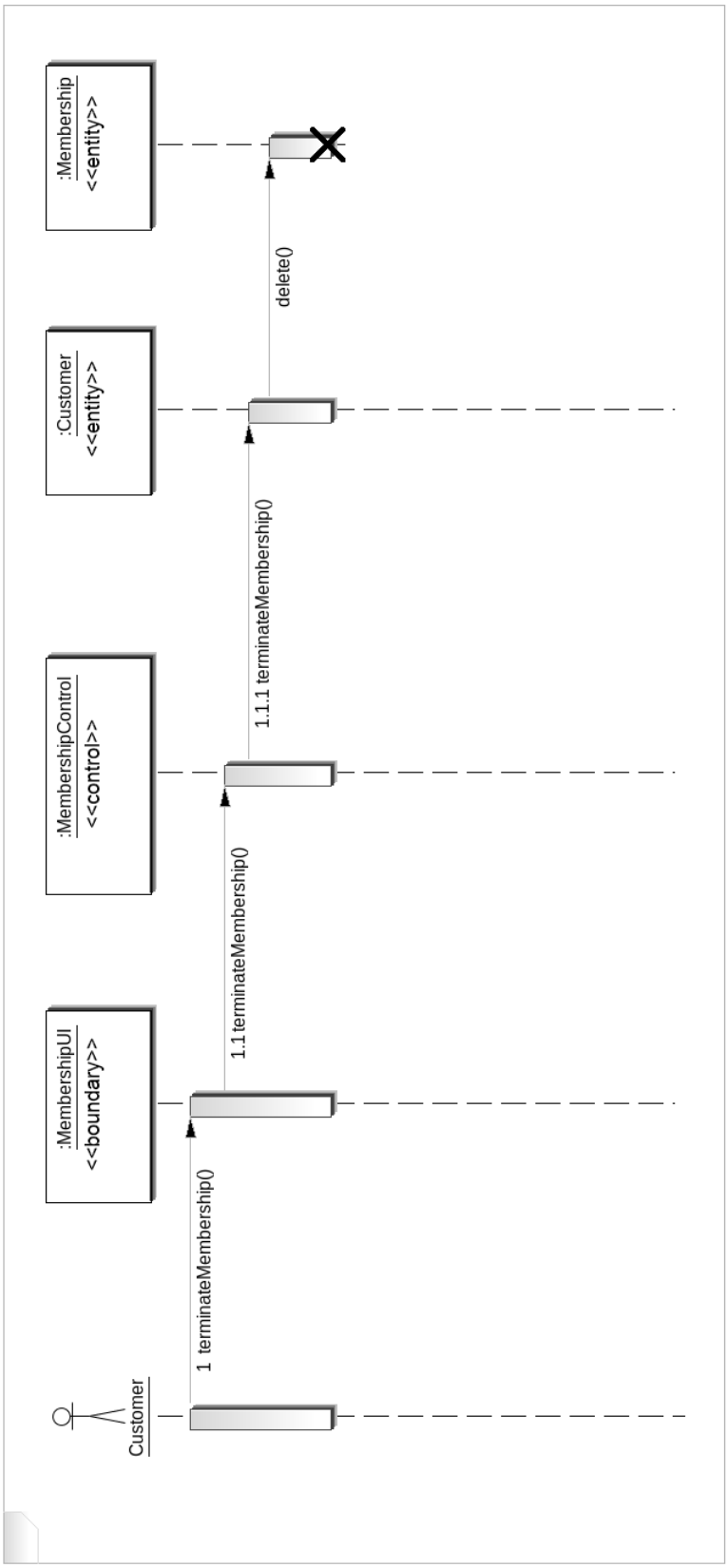
2.4.4.19 RemoveEmployee – Sequence Diagram – Use Case: Page 43



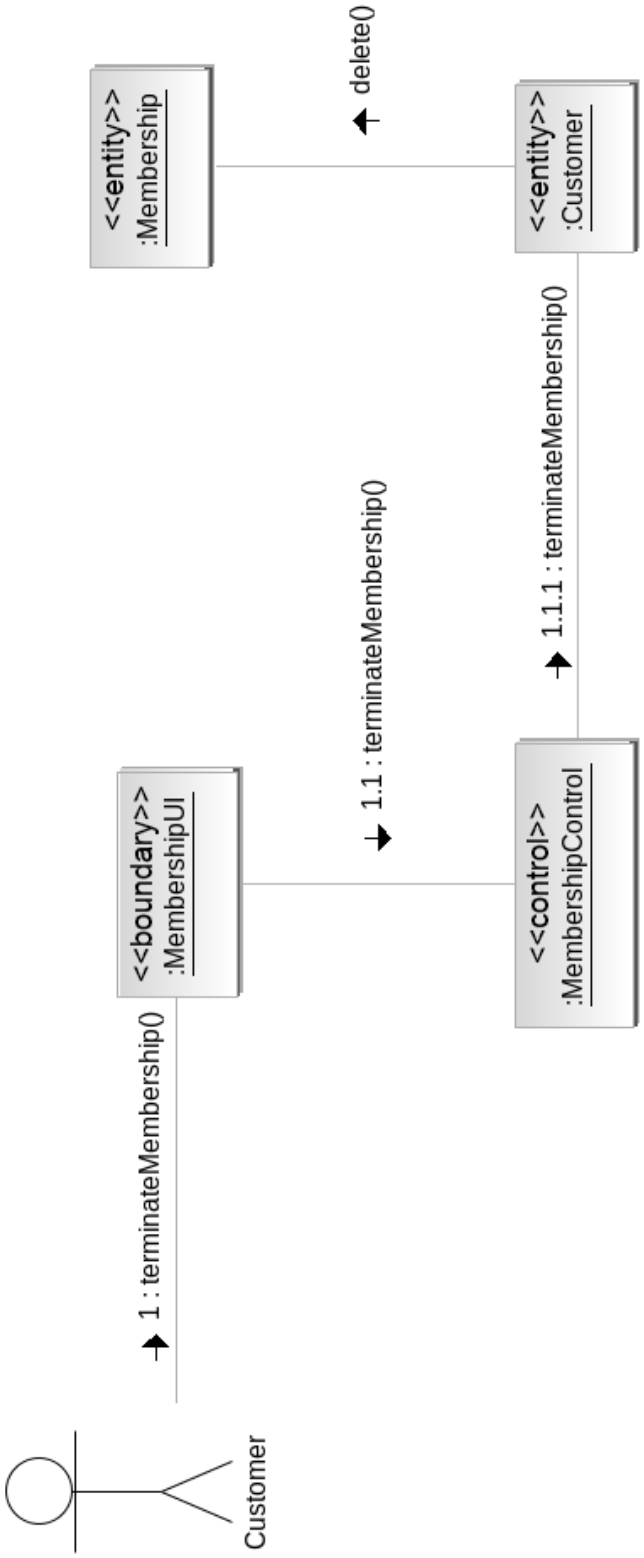
2.4.4.19 RemoveEmployee – Communication Diagram – Use Case: Page 43



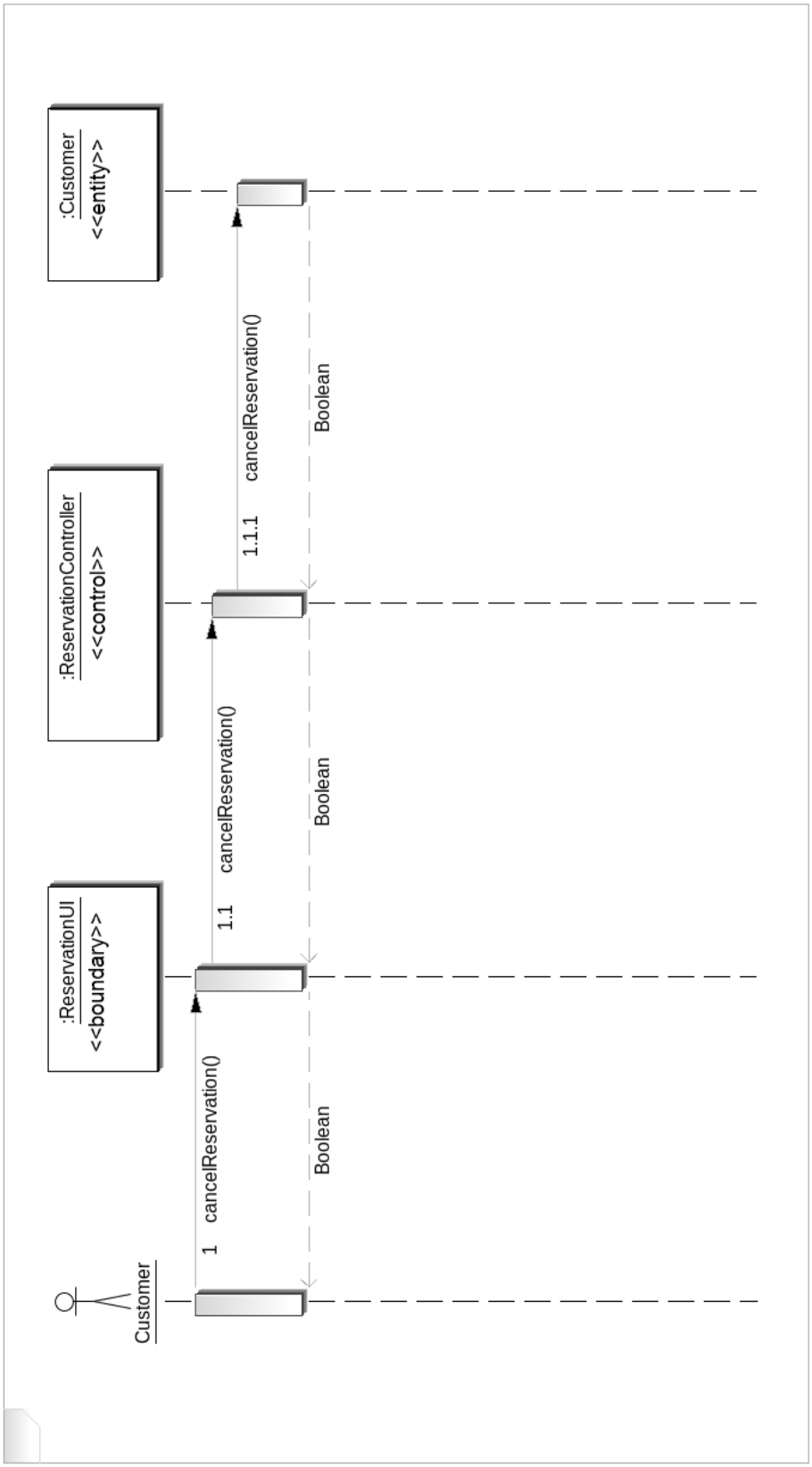
2.4.4.20 TerminateMembership – Sequence Diagram – Use Case: Page 44



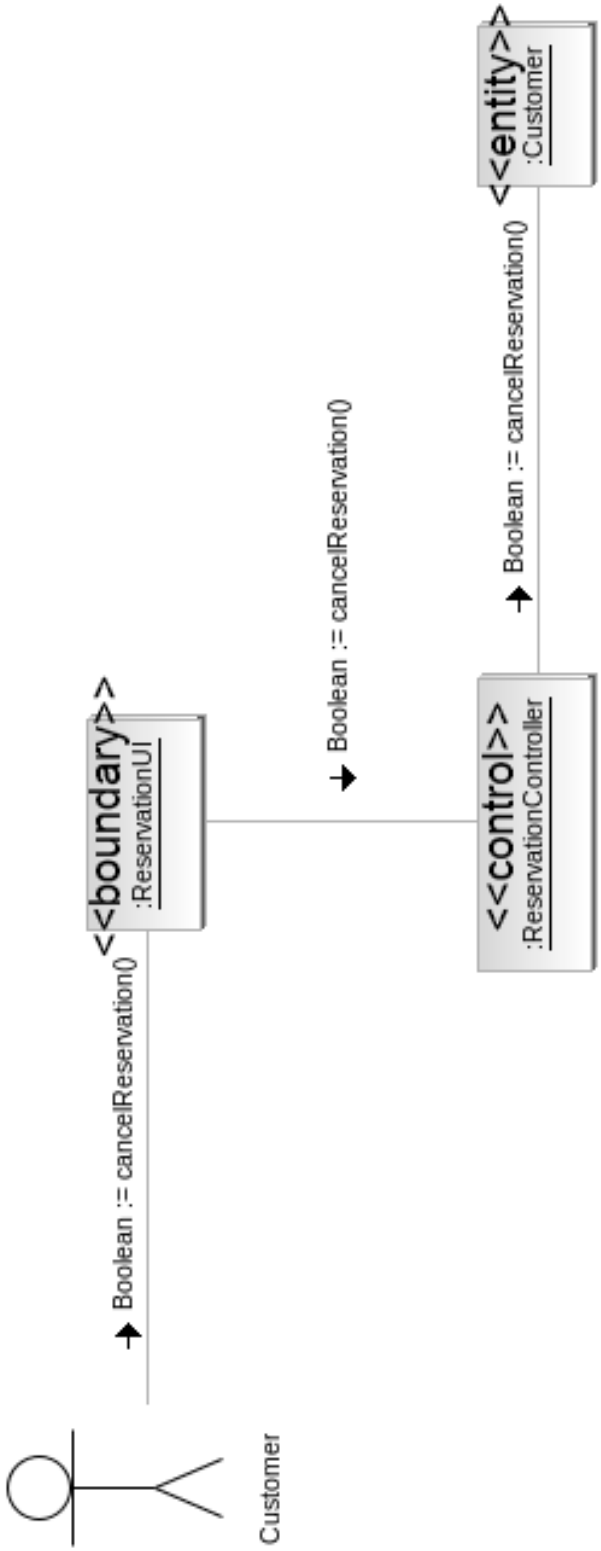
2.4.4.20 TerminateMembership – Communication Diagram – Use Case: Page 44



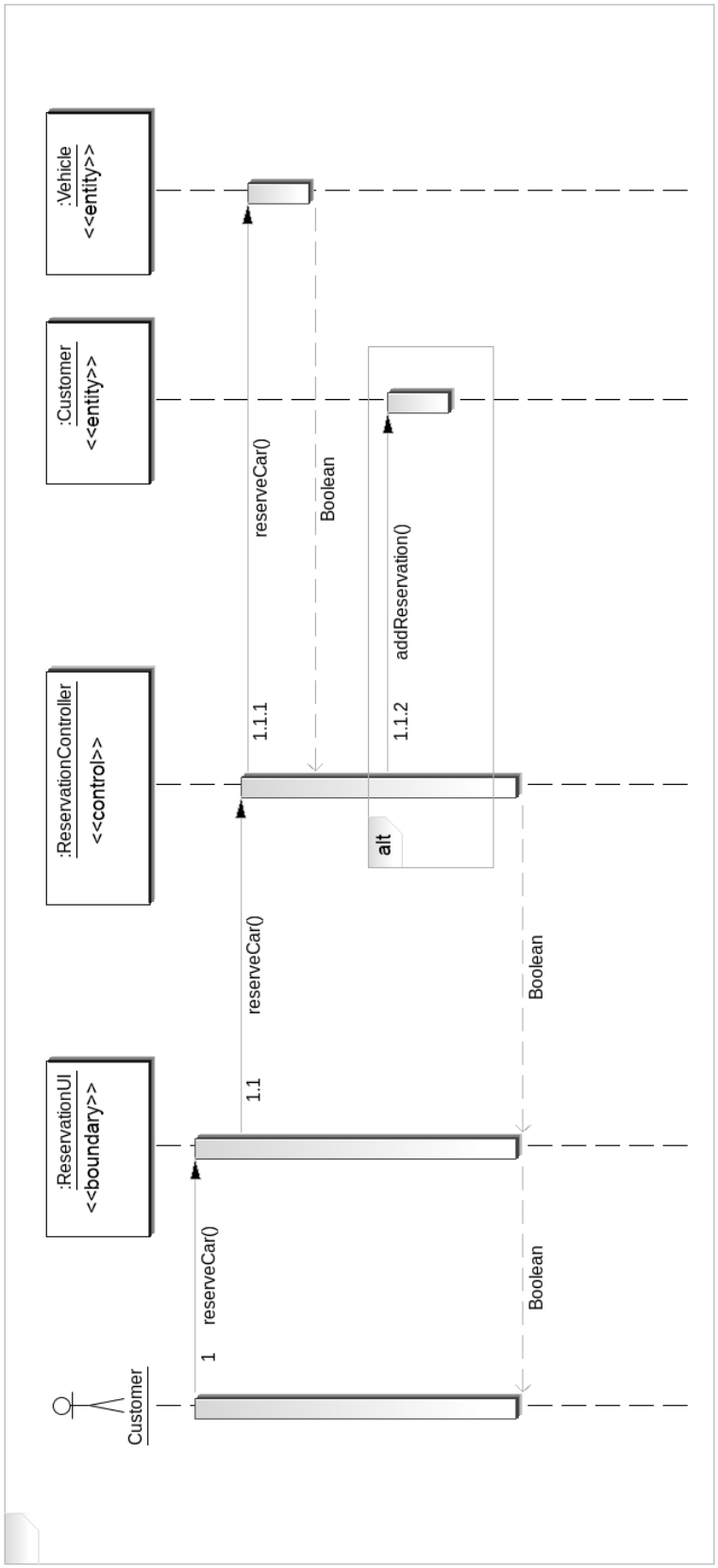
2.4.4.21 CancelReservation – Sequence Diagram – Use Case: Page 45



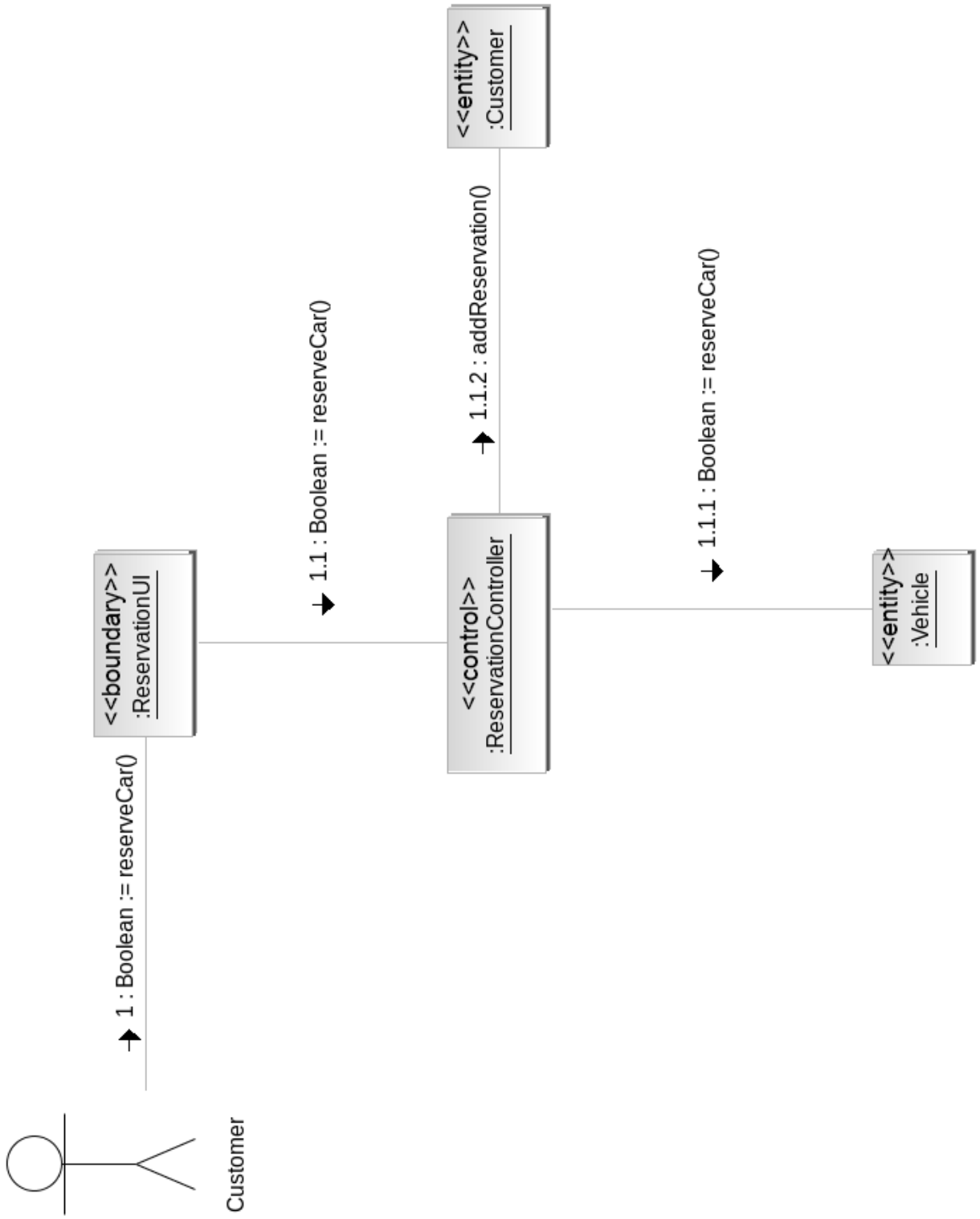
2.4.4.21 CancelReservation – Communication Diagram – Use Case: Page 45



2.4.4.22 ReserveCar – Sequence Diagram – Use Case: Page 47

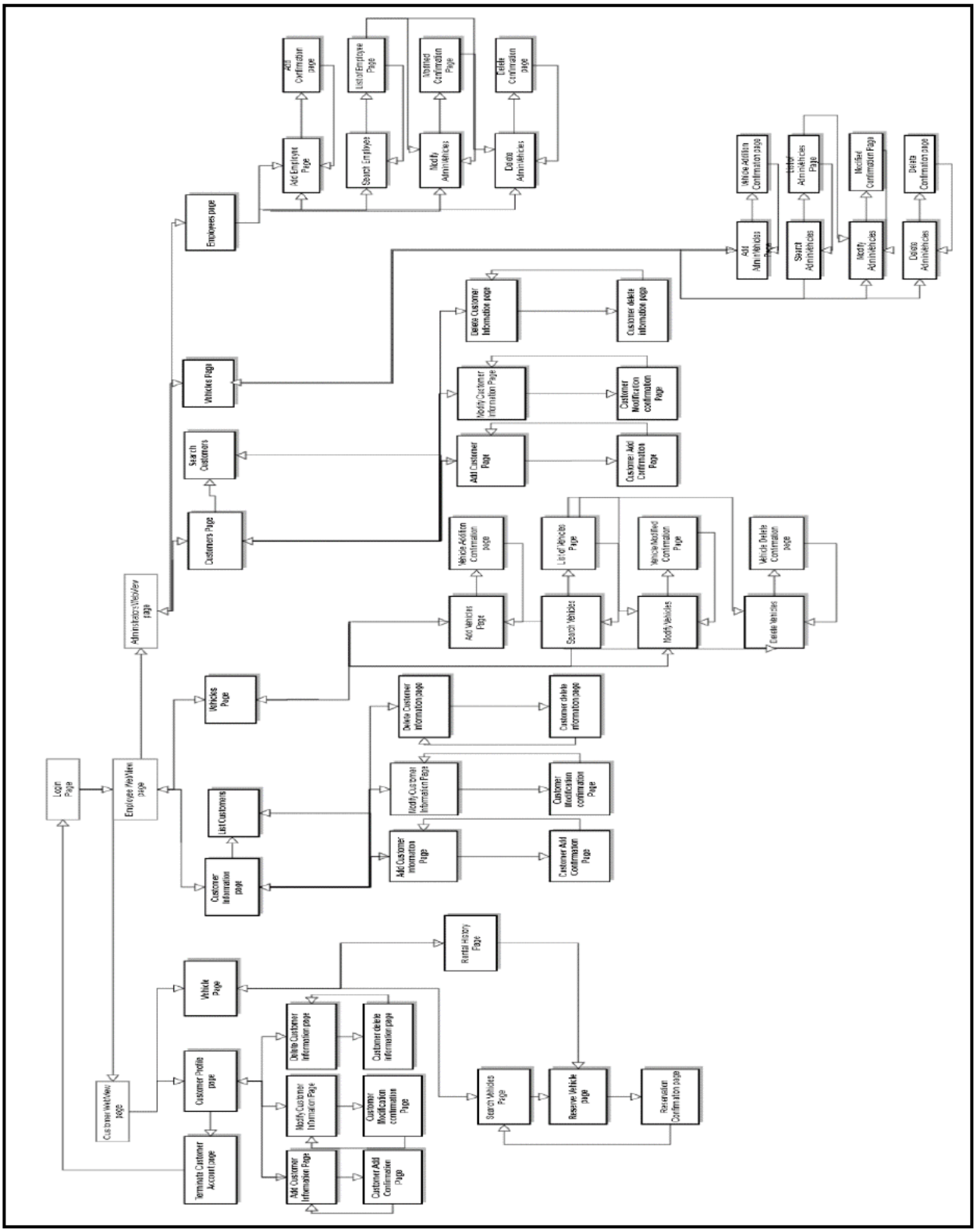


2.4.4.22 ReserveCar – Communication Diagram – Use Case: Page 47



2.4.5 Navigation path and User Interface

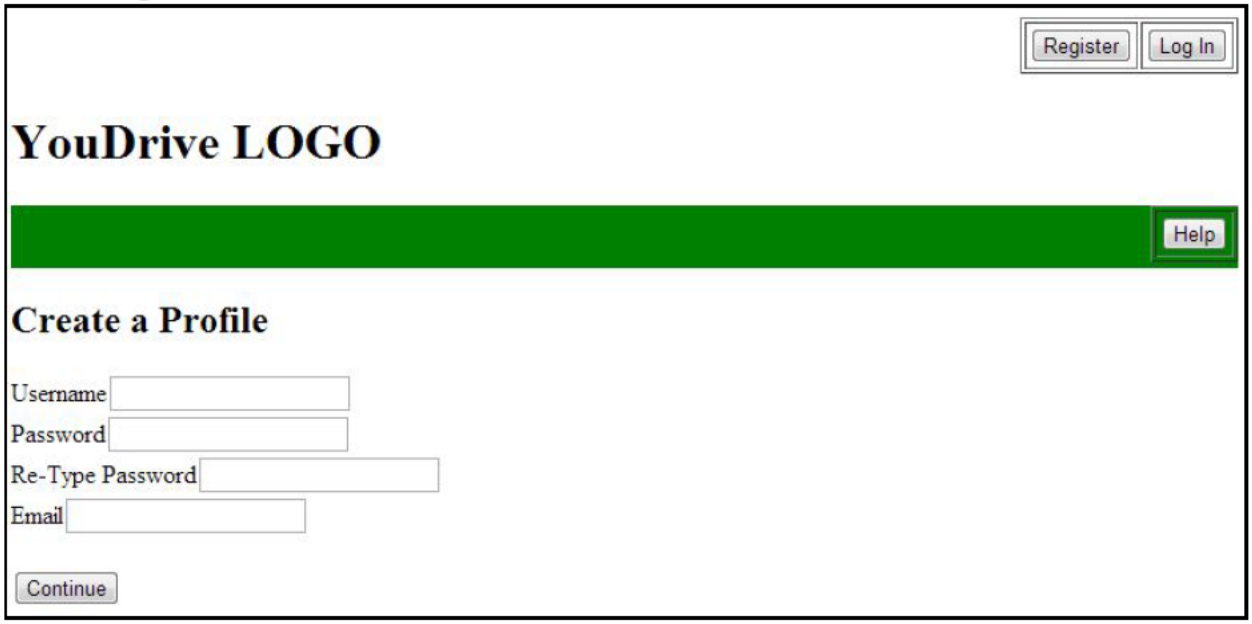
2.4.5.1 Navigation path



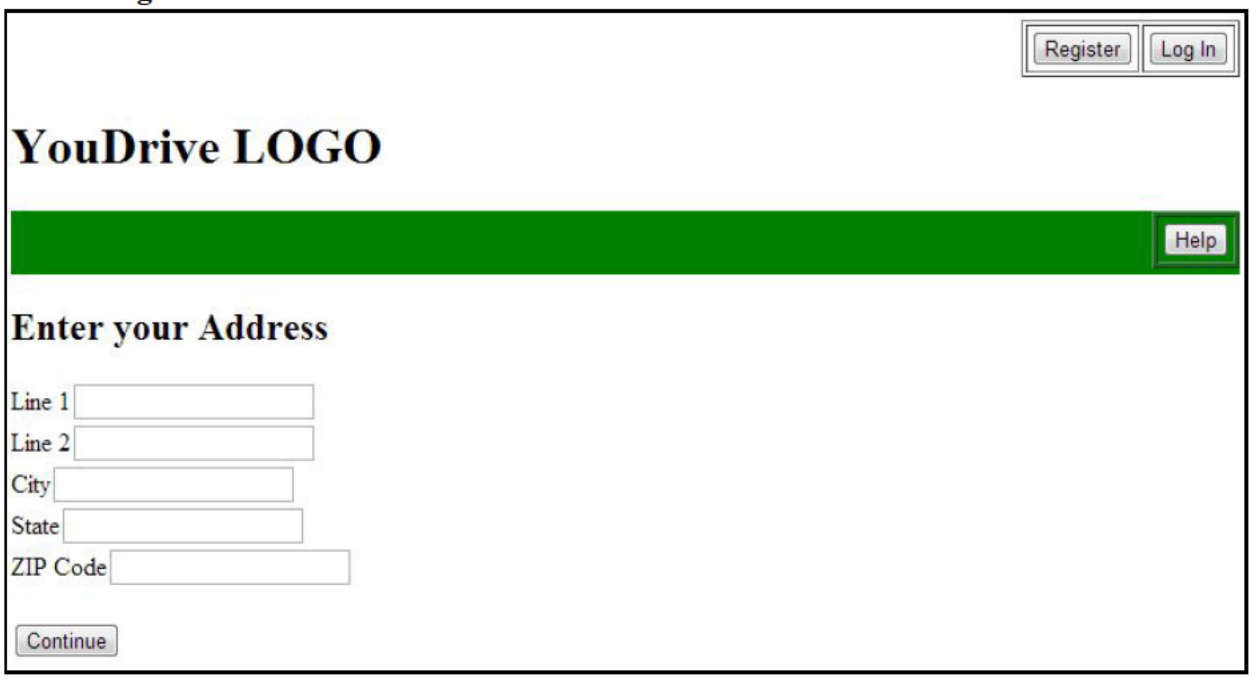
2.4.5.2 User Interface Mock-Up

The following user Interface shoots are a mock-up of the basic standards of how the system begin to be developed on a front end level. We have presented a sequential user interface mock up 1-14 illustrating an idea of the basic systems functions displaying how they will feel and operate.

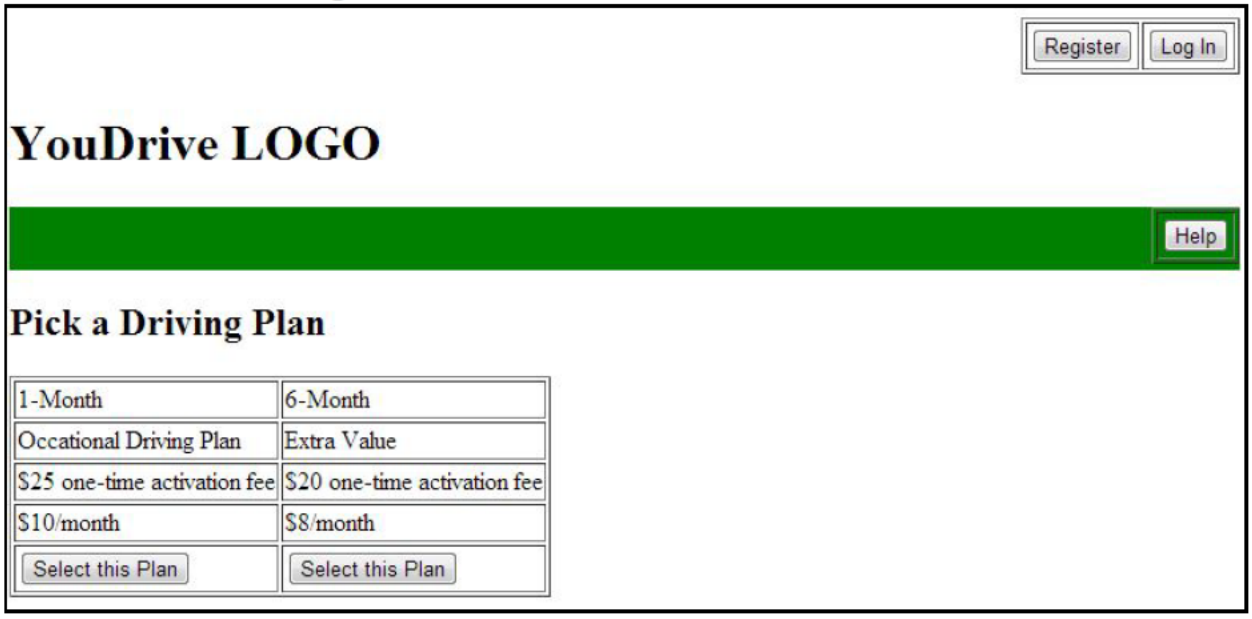
2.4.5.3 Registration of User Profile



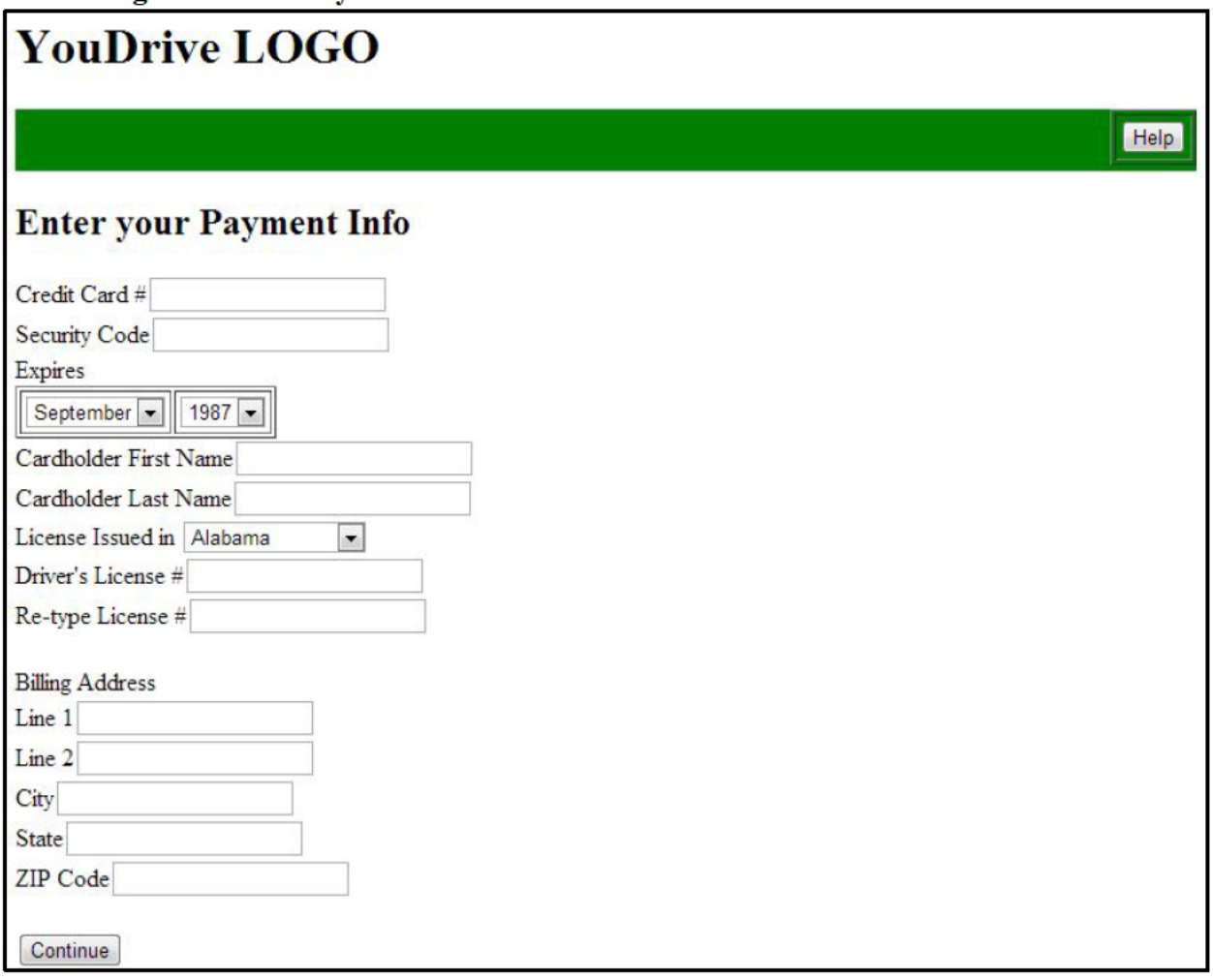
2.4.5.4 Registration of User Address Information



2.4.5.5 Choose of Driving Plans

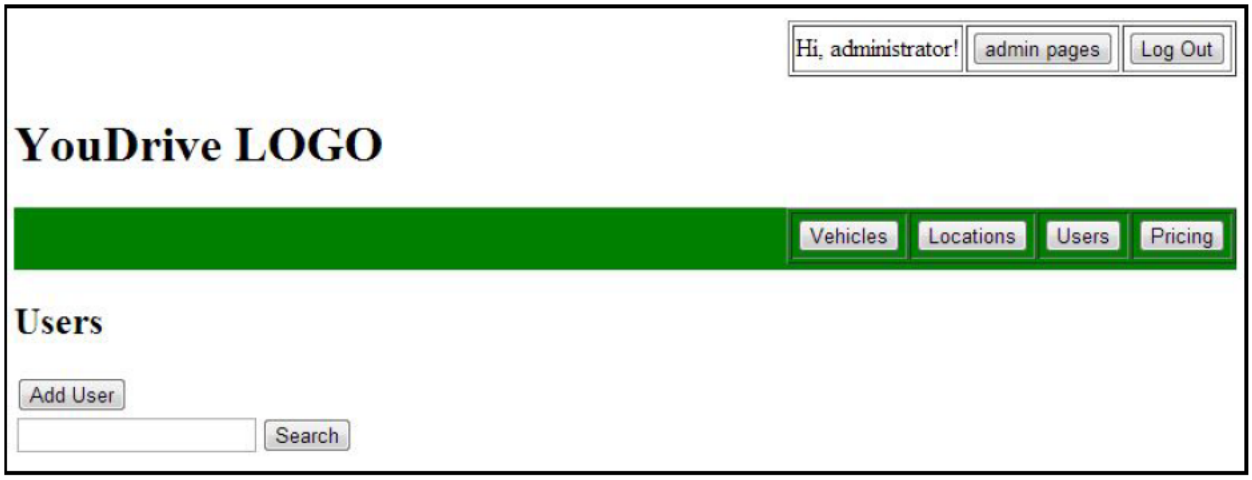


2.4.5.6 Registration of Payment Information



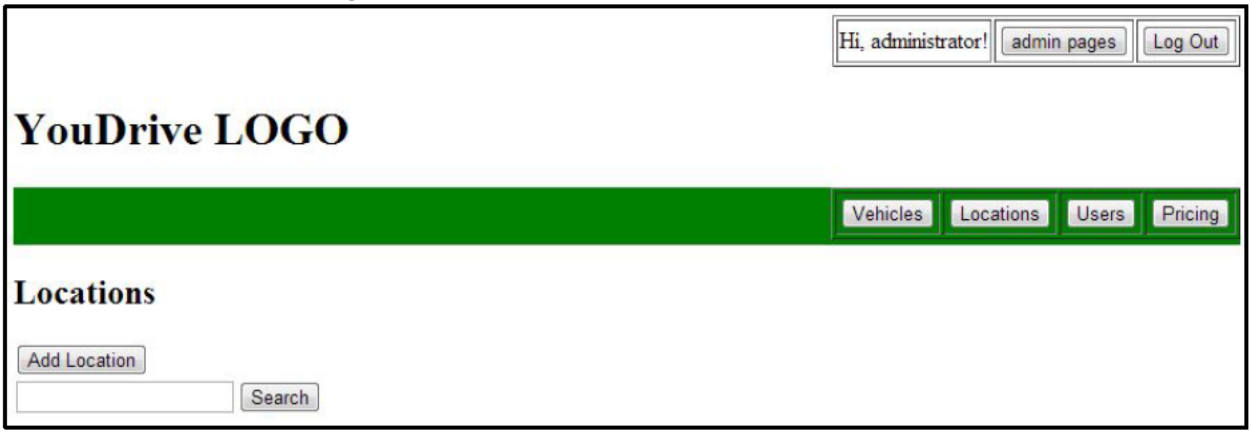
2.4.5.7 Sign Up Confirmation

2.4.5.8 AddUser

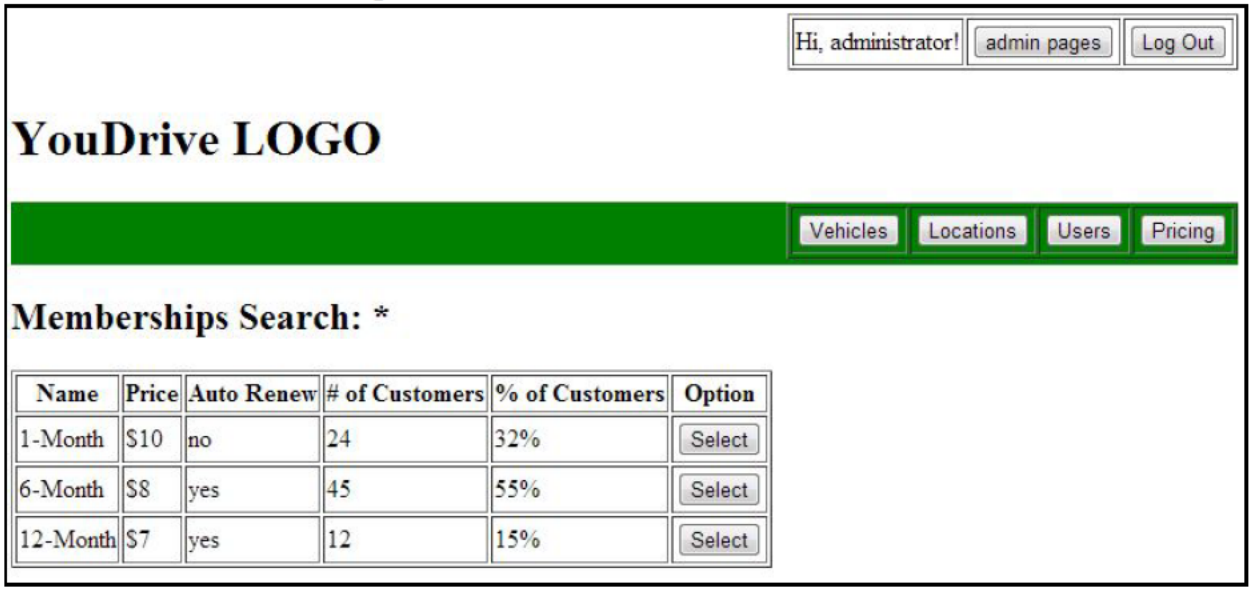


2.4.5.9 AddVehicles

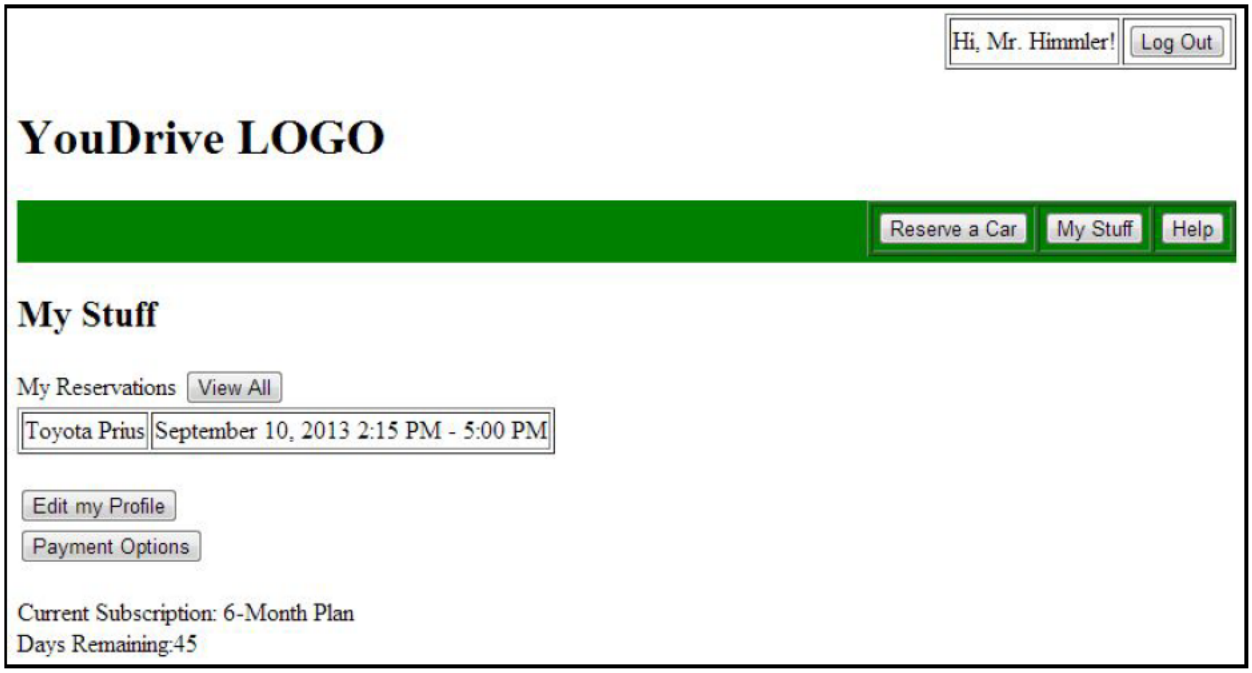
2.4.5.10 Search Vehicle By Location



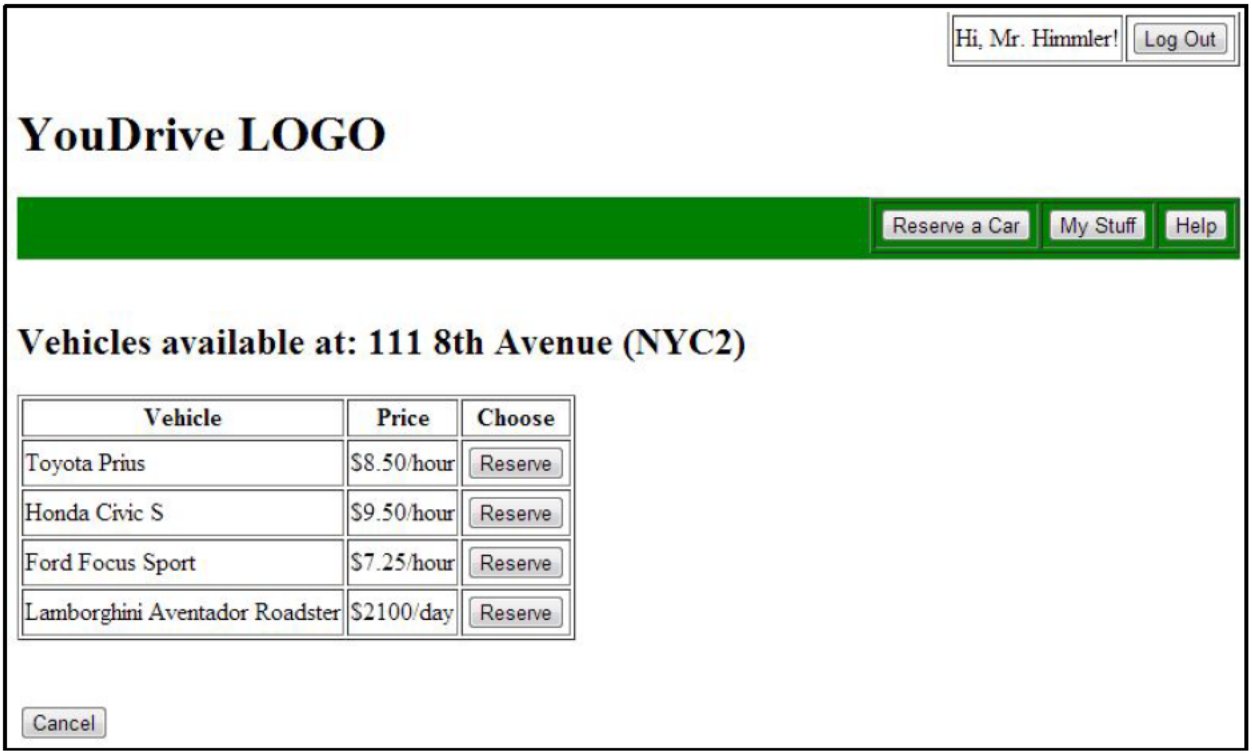
2.4.5.11 Admin Membership Search



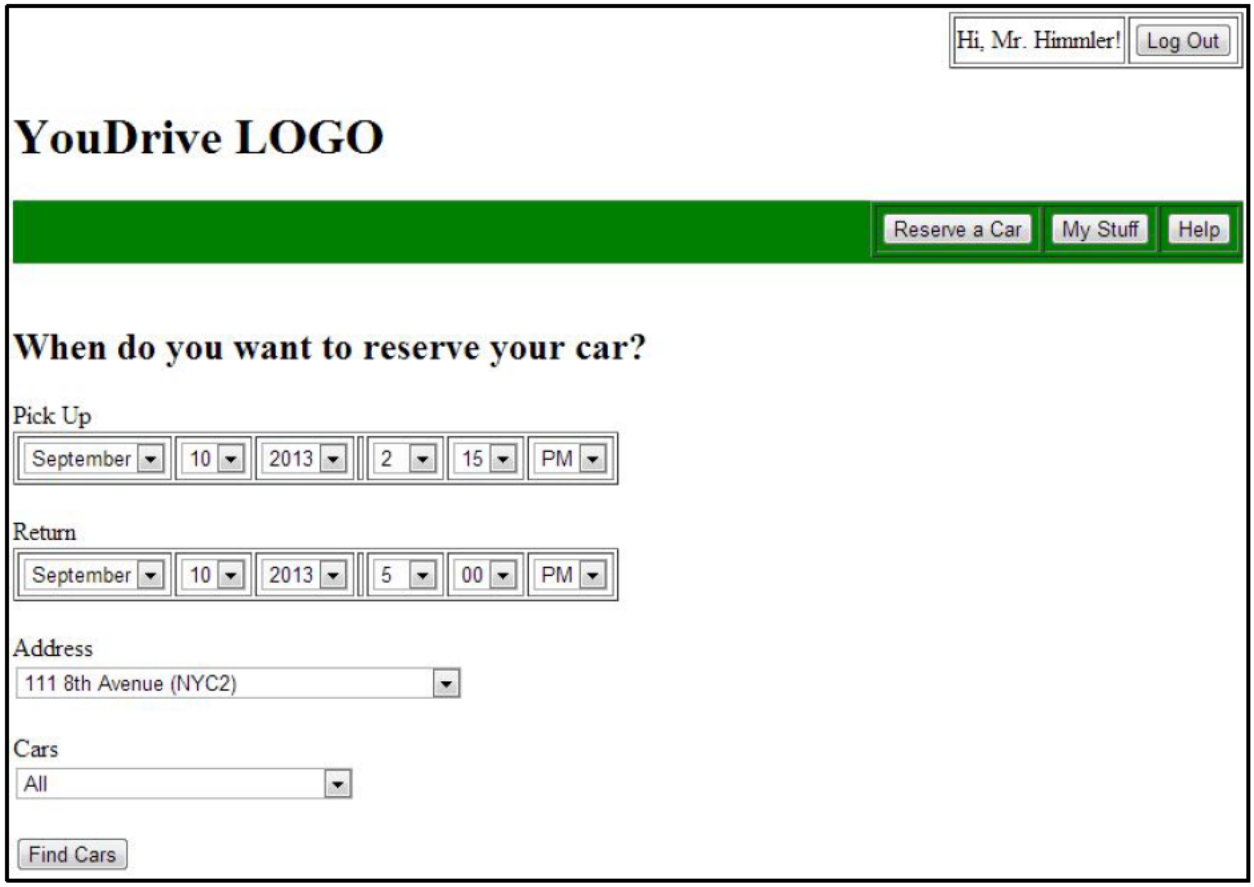
2.4.5.12 CustomerView



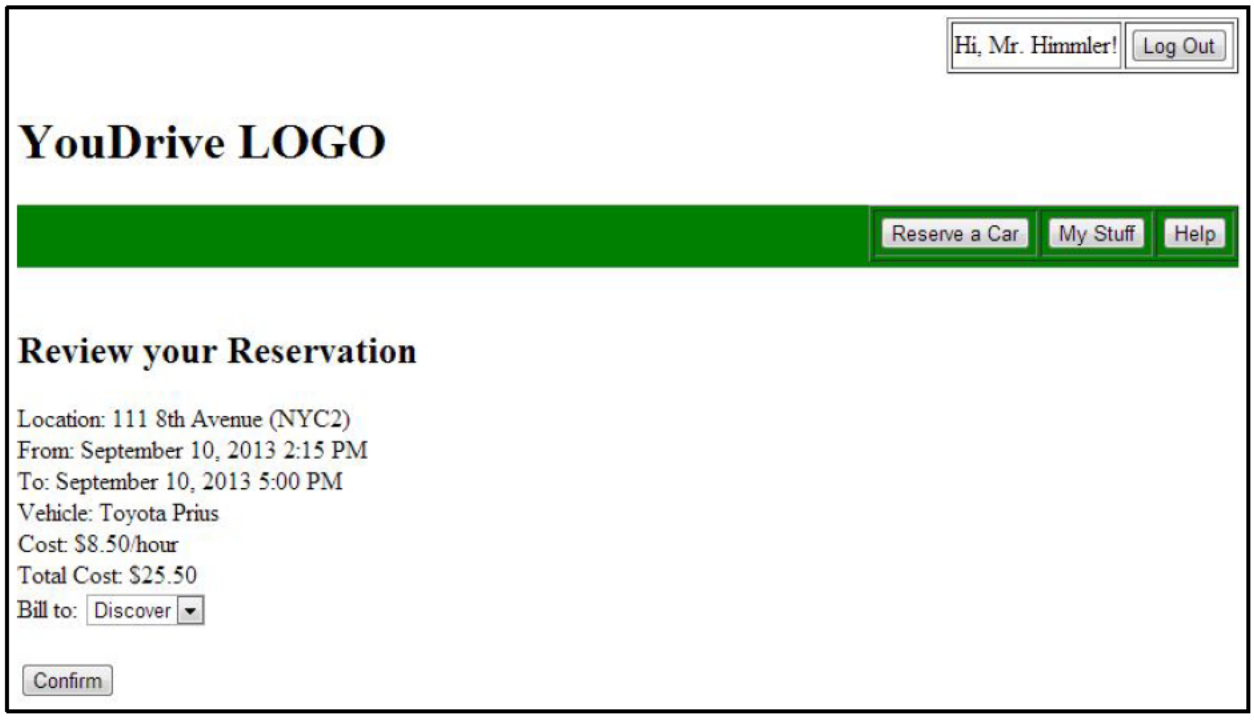
2.4.5.13 Vehicle Search



2.4.5.14 Car Reservation



2.4.5.15 Car Reservation Information



2.4.5.16 Reservation Confirmation