Content

Appendix A – Personally evaluations

Nicolaj Nyvang

As I am "the new guy" in the class, I was quite unsure of the group I chose (and visa versa). But with some good luck, theories about teams, and common sense the group grew to a team and I felt that the barrier that often can come when starting a new place wasn't there at all. This is one of the reasons that I have learned as much as I feel I did in this project.
The dedication among the others is in such a high degree which I haven't experienced in any of my former educations or jobs… This was the perfect place for me as I don't see this education as just an education. For me this is what I like doing in my spare time, and therefore my dedication is just as high.

So, what have I learned?
At the beginning, I was very confused because my new class had learned some things in a completely different way than myself. For example the order in which the domain model should be created. I have learned that the business analysis (which included the domain model) should be done before anything else in a report. No wonder that the others looked funny at me, when I proclaimed that we were doing it all wrong…

This however I got a hold on as we were advancing throughout the project and I feel that I am a little "richer" on knowledge, because I now know two somewhat different ways of doing a software project, and I can therefore better see the pros and cons when holding them up against each other.

In the beginning we agreed on exploring new features in java and if relevant, use these in our report/system. This has been a very fun way to learn new things in programing. I have done several of the special features like PDF creation (print order from the todolist), JavaMail, JavaHelp, and JavaFX. These are really nice features that can be very useful in many projects to come, and I often ask myself if I would have explored them if we didn't make this agreement. I properly would have, but definitely not at this point.

The most fun part was to make the PDF file, using a library called iText. This was really a challenge because I wanted to make a logo mark that should serve as a watermark on every document created. I had many difficulties trying to make the table lines disappear, and I have yet to find the solution.

A new thing this semester is the databases, and how to implement these. This is really a subject that I like, but this is also very challenging as the logic is quite different from the different areas I have explored before. I felt kind of lost after the classes because of some unlucky coincidences with sickness (both wife and kid) at home which didn't give me the time that I felt I needed. Fortunately I was persistent and I read in several different books and as the finishing touch I attended a course in SQL before working with the practical part regarding SQL in our project.
Another new thing I discovered this semester was the possibility for watching video tutorials about both java and UML. This is a very good way to supplement the reading, as another angle of things can often help me understand things in a whole new way.
My primarily role in this project were project manager as this role comes very naturally to me, and I actually have a hard time leaving the hat at home.. But I feel that I have given the space needed for the others to try the role as well – I hope they agree ☺

Finally I just want to say that this semester has been the most instructive for me, as I feel that I really have learned a great deal in our subjects.

- Nicolaj Nyvang

Adams Mukubi

First and foremost I would like to thank my teachers (Paul and Susana) who have helped me a lot for the completion of my 2nd semester and my group members because some of them have really bean of great help as well to some extent. The project has been quite big and my group members have been so ambitious to implement a lot of things which is positive to me because I came to learn about a lot of new things. Though to some extent I could feel like there was some difference in culture between myself and some group members but I accepted that in order to come up with a common goal, on top of that my mind have not been so stable because of some personal problem. All in all it has been a nice project with a lot to do and in a short period.

Claus Pallisgaard Beck

It has been exciting to be a part of a project, there you have the opportunity to try new features and get response from rest of the group. It have also been good that the project is part of the “real life”, this make it easier to work toward a final result that someone could use in where company.

I have got an idea and some hints to my further practice in computer science and how to make projects. The recipe to make it easier to myself.

I have got a great view into SQL and JAVA JDBC, it have been a great experience, and very useful for my further education.

I’m looking forward to end this project as a final and working system, and deliver it to the company.

Torben Hansen

As the project grew larger and larger i realized that at some point i don't need to know everything about every single line of code or every diagram. Since this project is beyond 35000 lines of code, i must satisfy myself with the fact that if i need insight in classes made by others, i could get it just by spending some time studying it. Here the diagrams come in handy.

We have just cleaned up the code and it was a pleassure to see that java-doc and the code-standard was applyed to nearly everything. Since our code is written in the same way, it's much easier to read after a small amount of time.

One of my conclusions is: It's sure no problem to write code that the computer understands, but it's much more complicated to write it for others.

I can also conclude that i should spent some time on talking with my team mates when i feel that i don't deliver as much as i though i should. Sometimes it's just a wrong perception which i have, and that isn't helpful.

It's also important to mension, that my team gave me the right challenge be aiming high and being sympathetic on the personal level.

Lars Schou

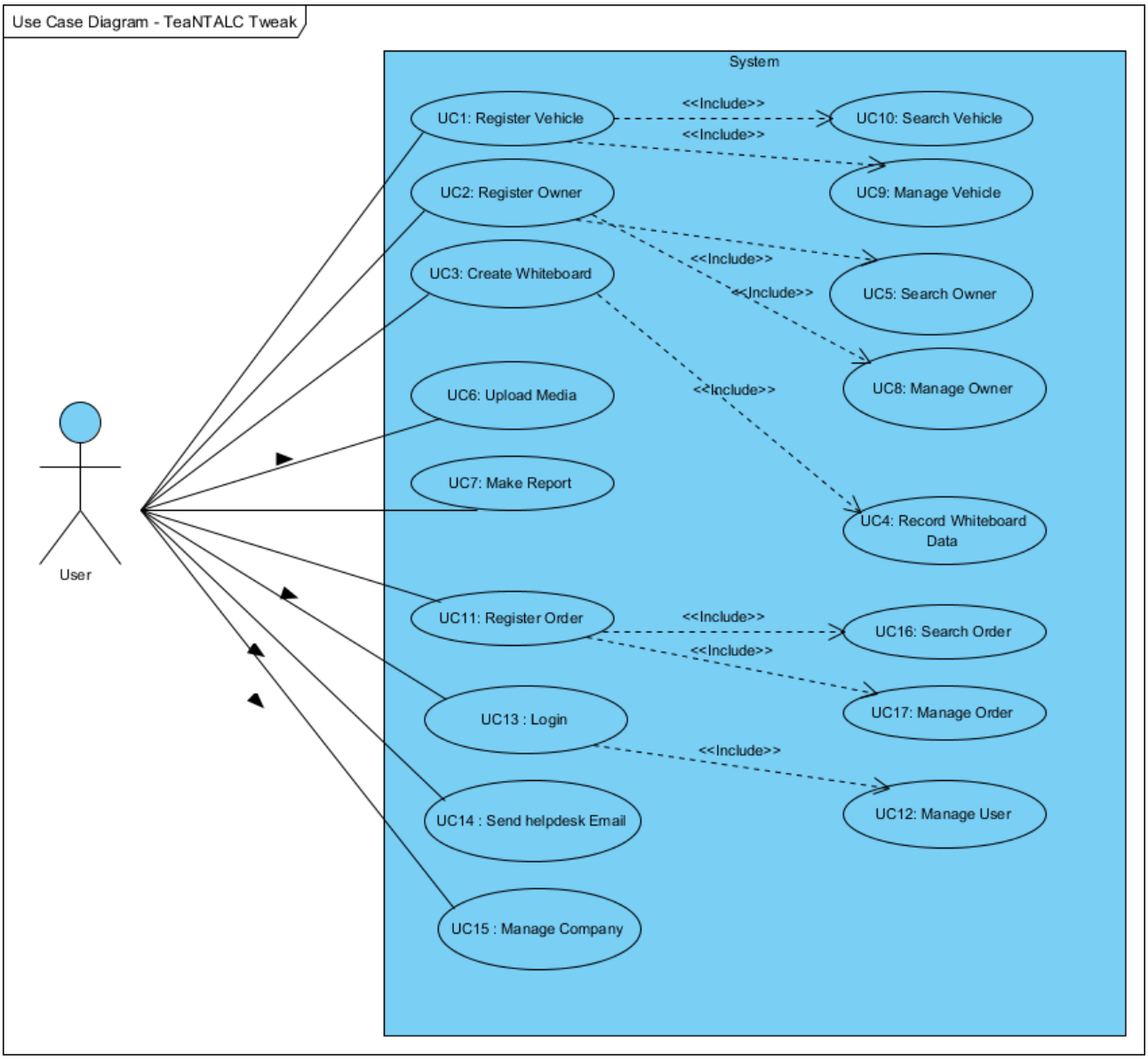
Throughout the project I've learnt of course a lot of theory and practical about UP and the Java language. This was though expected as I had set myself a very big personal goal on these topics.
Anyway I've learned things I didn't expect. As the first I've learned that, being in a team is an incredibly personally resource-intensive work for all team members to get everything to run. Furthermore I've learned that to plan the work to be done while keeping the schedule is very difficult and probably requires much experience. I've also learned that to maintain a good team, as well as the team spirit it is important that everyone feels like a part of the team, so that everyone says their opinions and views on things when there are discussions. As the last I will write on the personal part, I would say that I certain feel I'm in a team working as a team should, where 2 +2 = 5 ☺

Appendix B – Use Case

Full appendix use cases – See All use cases at the end of this appendix.

Fully dressed use cases

Use-case diagram



UC2: Register Owner

Name	Value
Name	Register Owner
Use Case ID	2
Rank	High
Primary Actors	User

Flow of Events

1.Need for owner registrations occurs
2. User starts new owner registration
3. User enters basic owner information(firstName, lastName, address, address2, zip, country, mail, phone)
3.1. User choose to attach owner to existing vehicle
3.2. User start Search Vehicle
3.3. User attach owner to selected vehicle
4.System gives ownerID
5.System saves owner
6. User ends owner registration

Extension

2.a. At any time system fails:
1. User starts a new ownerregistration
3.2.a. Vehicle doesn't exist
1. User start Register Vehicle
2. User goes to step 3.1
3.a. User types in wrong information
1. User resets fields and try again
5.a. System fails to save owner
1. User enters all basic information again
2.System saves owner
6.a. At any time system send error message:
1. User press F1 and are shown Help Menu

2.  User read and tries understanding information in Help Menu
2.1.  User do not understand Help Menu
2.1.1.  User sends an email via the helpdesk function
2.2.  User understands and goes to step 3
3.  User tries again

Details

Name	Value
Level	User Goal
Complexity	Easy
Use Case Status	Finish
Implementation Status	Finish
Preconditions	User is logged in and all information about a owner is available
Post-conditions	Information for Owner is saved
Author	NegoZiatoR

UC3: Create Whiteboard

Name	Value
Name	 Create Whiteboard
Use Case ID	3
Rank	High
Primary Actors	User

Flow of Events

1.Need for creating whiteboard occurs
2.  User chooses whiteboard type
3.System creates whiteboard with rows and columns from users selected type
4.System present whiteboard.

Extension

2.a. At any time system fails:
1.  User starts a new create whiteboard
2.b.  User types in wrong information
1.  User resets fields and try again
3.a. System fails to create whiteboard
1.  User saves work and restarts system
2.System start up again
2.1.System fails again to create
2.1.1.  User restarts computer and goes to step 1 again
4.a. At any time system send error message:
1.  User press F1 and are shown Help Menu
2.read and tries understanding information in Help Menu
2.1.  User do not understand Help Menu
2.1.1.  User sends an email via the helpdesk function
2.2.  User understands and goes to step 3
3.  User tries again

Details

Name	Value
Level	User Goal
Preconditions	Vehicle is selected A WhiteboardCatalog Exist
Post-conditions	Whiteboard is created
Author	NegoZiatoR
Frequency of occurrence:	Approximately about 400 times (4 whiteboards pr engine done)

UC4: Manage Whiteboard

Name	Value
Name	Manage Whiteboard
Use Case ID	4
Rank	High
Primary Actors	User

Flow of Events

1.Need for manage whiteboard occurs
2. User starts manage whiteboard
3. User enters /edit data
4.System do calculations if necessary
5.System saves whiteboard
6. User ends manage whiteboard

Extension

2.a. At any time system fails:
1. User starts a new record measurements
3.a. User types in wrong information
1. User resets fields and try again
5.a. System fails to save
1. User saves work and restarts system
2.System restart
2.1.System fails to save again
2.1.1. User restarts computer and goes to step 1 again
3. User saves data
6.a. At any time system send error message
1. User press F1 and are shown Help Menu
2. User read and tries understanding information in Help Menu
2.1. User do not understand Help Menu
2.1.1. User sends an email via the helpdesk function

2.2.  User understands and goes to step 3
3.  User tries again

Details

Name	Value
Level	User Goal
Preconditions	Correct whiteboard is present A WhiteBoardCatalog exist
Post-conditions	Whiteboard is saved
Author	NegoZiatoR
Frequency of occurrence:	As many times a whiteboard is created, approximately 400 times

UC5: Search Owner

Name	Value
Name	 Search Owner
Use Case ID	5
Rank	Medium

Flow of Events

1.Need for owner search occurs
2.  User starts a new owner search
3.  User enters search information (ownerID || name || phone || address || mail)
4.System shows search result (Owner || Null)
5.  User ends search owner

Extension

2.a. At any time system fails
1. User starts a new owner search
3.a. User types in wrong information
1. User resets fields and try again
4.a. System fails
1. User restarts system
2.System starts again
2.1.System fails again
2.1.1. User restarts computer and goes to step 1
3.System shows found owner
5.a. At any time system send error message
1. User press F1 and are shown Help Menu
2. User read and tries understanding information in Help Menu
2.1. User do not understand Help Menu
2.1.1. User sends an email via the helpdesk function
2.2. User understands and goes to step 3
3. User tries again

Details

Name	Value
Level	User Goal
Complexity	Easy
Use Case Status	Finish
Implementation Status	Finish
Preconditions	User is logged in, Owner is present and all information available
Post-conditions	Information for owner is returned to the User
Author	Nyvang
Frequency of occurrence:	Approximately 1000 times pr year

UC6: Upload Media

Name	Value
Name	 Upload Media
Use Case ID	6
Rank	Medium
Primary Actors	User

Flow of Events

1.Need for upload media occurs
2.  User starts upload session
3.  User selects media type and location
4.System attach media to vehicle
5.  User ends upload session

Extension

2.a. At any time system fails
1.  User starts new upload media
5.a. At any time system send error message
1.  User press F1 and are shown Help Menu
2.  User read and tries understanding information in Help Menu
2.1.  User do not understand Help Menu
2.1.1.  User do still not understand
2.1.1.1.  User quits and goes home
2.2.  User understands and goes to step 3
3.  User tries again

Details

Name	Value
Level	User Goal
Complexity	Medium
Preconditions	Picture is taken, sound is recorded, or video is taken. Vehicle is selected
Post-conditions	Media is attached to vehicle.
Author	Adams
Frequency of occurrence:	Approximately 750 times pr. year

UC7: Make Report

Name	Value
Name	 Make Report
Use Case ID	7
Rank	High
Primary Actors	User

Flow of Events

1.Need for report occurs
2.  User start Make Report
3.  User choose report template(owner or workshop)
4.  User User choose type of report (screen or paper or moveable media or a combination of these 3)
5.  User choose which element to be in report(image, sound, videos, stations, descriptions, notes(intern/extern) (choice is restricted by the report type selected)
6.System generate report
7.  User ends Make report

Extension

2.a. At any time system fails
1.  User starts new Make Report
2.b. At any time system send error message
1.  User press F1 and are shown Help Menu
2.  User read and tries understanding information in Help Menu
2.1.  User do not understand Help Menu
2.2.  User call Hotline
2.2.1.  User do still not understand
2.2.1.1.  User quits and goes home
2.3.  User understands and goes to step 3
3.  User tries again
4.a.  User does not have a printer device or moveable media
1.  User choose Show on screen
6.a. System can’t connect to output device
1.System restarts connection and tries again
1.1.system can’t still no connect to output device
1.1.1.  User must choose another output
6.b. Printer is out of paper
1.  User must fill paper
1.1.  User has no paper
1.1.1.  User must choose another output
6.c. System missing moveable media
1.  User attach moveable media
1.1.  User does not have a moveable media
1.1.1.  User must choose another output
2.  User tries again

Details

Name	Value
Level	User Goal
Complexity	Hard
Preconditions	Vehicle is registered in the system Vehicle is selected
Post-conditions	Report is printed or shown or put on moveable media or combination of these 3
Author	Claus Beck
Frequency of occurrence:	1500 times pr year

UC8: Manage Owner

Name	Value
Name	 Manage Owner
Use Case ID	8
Rank	Low

Flow of Events

1.Need for manage owner occurs
2.  User start manage owner
3.  User choose manage type
4.  User delete or updates owner
5.System updates data
6.  User ends manage owner

Extension

2.a. At any time system fails
1.  User starts new manage owner
2.b. At any time system send error message
1.  User press F1 and are shown Help Menu
2.  User read and tries understanding information in Help Menu
2.1.  User do not understand Help Menu
2.1.1.  User sends an email via the helpdesk function

2.1.1.1.  User do still not understand
2.1.1.1.1.  User quits and goes home
2.2.  User understands and goes to step 3
3.  User tries again

Details

Name	Value
Level	User Goal
Complexity	Easy
Use Case Status	Finish
Implementation Status	Finish
Preconditions	Owner is selected
Post-conditions	Owner is updated or deleted
Author	NegoZiatoR
Frequency of occurrence:	Approximately 200 times pr year

UC9: Manage Vehicle

Name	Value
Name	 Manage Vehicle
Use Case ID	9
Rank	Low

Flow of Events

1.Need for manage vehicle occurs
2.  User start manage vehicle
3.  User start  Search Vehicle
4.  User Select found vehicle
5.  User choose manage type
6.  User delete or update vehicle
7.System updates data
8.  User ends the manage vehicle

Extension

2.a. At any time system fails
1.  User starts a new Manage Vehicle
6.a.  User types in wrong information
1.  User resets fields and try again
7.a. System fails to save
1.  User restarts system
2.System restarts
2.1.System fails to save again
2.1.1.  User restarts computer and goes to step 1
3.System shows found vehicle
3.1.  User goes to step 4
8.a. At any time system send error message
1.  User press F1 and are shown Help Menu
2.  User read and tries understanding information in Help Menu
2.1.  User do not understand Help Menu
2.1.1.  User sends an email via the helpdesk function
2.2.  User understands and goes to step 3
3.  User tries again

Details

Name	Value
Level	User Goal
Use Case Status	Finish
Implementation Status	Finish
Preconditions	Vehicle is selected
Post-conditions	Vehicle is updated or deleted.
Author	Nyvang
Frequency of occurrence:	Approximately 300 times pr. year

UC10: Search Vehicle

Name	Value
Name	 Search Vehicle
Use Case ID	10
Rank	Medium

Flow of Events

1.Need for Vehicle search occurs
2.  User starts a new Vehicle search
3.  User enters search details (vehicleID || brand || licensePlate || chassisnumber)
4.System shows search result (Vehicle | | Null)
5.  User end Vehicle search

Extension

2.a. At any time system fails
1.  User starts a new Search Vehicle
3.a.  User types in wrong information
1.  User resets fields and try again
5.a. At any time system send error message
1.  User press F1 and are shown Help Menu
2.  User read and tries understanding information in Help Menu
2.1.  User User do not understand Help Menu
2.1.1.  User sends an email via the helpdesk function
2.2.  User understands and goes to step 3
3.  User tries again

Details

Name	Value
Level	User
Complexity	Medium
Use Case Status	Finish
Implementation Status	Finish
Preconditions	Vehicle is present and all information available
Post-conditions	Vehicle information is shown
Author	NegoZiatoR
Frequency of occurrence:	Appoximately 1000 times pr year

UC11: Register Order

Name	Value
Name	Register Order
Use Case ID	11
Rank	Medium
Primary Actors	User

Flow of Events

1.Need for Order registration occurs
2. User starts a new Order registration
3. User enters basic Order information (vehicleID, orderType, description, startDate, expEndDate)
4.System give orderID
5.System saves order
6. User ends order registration

Extension

2.a. At any time system fails
1. User starts a new Vehicle registration
3.a. User types in wrong information
1. User resets fields and try again
5.a. System fails to save order
1. User enters all basic information again
2.System saves Vehicle
6.a. At any time system send error message
1. User press F1 and are shown Help Menu
2. User read and tries understanding information in Help Menu
2.1. User do not understand Help Menu
2.1.1. User sends an email via the helpdesk function
2.2. User understands and goes to step 3
3. User tries again

Details

Name	Value
Level	User Goal
Complexity	Medium
Preconditions	User logged on, vehicle and owner is present in system
Post-conditions	Order is registered
Author	Claus Beck
Frequency of occurrence:	When vehicle need to be repaired/optimized approximately 750 per year.

UC12: Manage User

Name	Value
Name	 Manage User
Use Case ID	12
Rank	Low

Flow of Events

1.Need for manage user occurs
2.  User starts a new manage user session
3.  User chooses edit existing user or create new user
3.1.  User Choose Create new
3.1.1.System asks for initials for user to be created
3.2.  User Choose Edit existing
3.2.1.System asks for initials for user to be edited
4.System saves the user
5.  User ends Manage User

Extension

2.a. At any time system fails
1.System reconstructs prior state
3.1.a.  User already exists
1.  User ends manage user
4.a. System fails to save

1.  User restarts system
2.System restarts
2.1.System fails to save again
2.1.1.  User restarts computer and goes to step 1 again
3.  User starts new register user
5.a. At any time system send error message
1.  User press F1 and are shown Help Menu
2.  User read and tries understanding information in Help Menu
2.1.  User do not understand Help Menu
2.1.1.  User sends an email via the helpdesk function
2.2.  User understands and goes to step 3
3.  User tries again

Details

Name	Value
Level	User Level
Complexity	Easy
Preconditions	System is running and a main session is established A User Catalog exist
Post-conditions	A new user is registered or updated and saved
Author	Tvupper
Frequency of occurrence:	Approximately 50 times pr year

UC13: – Login (Missing)

UC14: Send helpdesk Email

Name	Value
Name	 Send helpdesk Email
Use Case ID	14
Rank	Low
Primary Actors	User

Flow of Events

1.Error symptoms is identified by  User
2.  User starts the help menu and selects "Report technical issue"
3.The symptoms are described into the textbox
4.System adds the log file as an attachment
5.E-mail is send from a predefined email address, to another predefined address
6.  User ends send helpdesk email

Details

Name	Value
Level	User Goal
Complexity	Hard
Preconditions	User has issues that cannot be resolved by self and therefore need to be escalated. Log file and internet connection must be available
Post-conditions	Error report has been send
Author	Nyvang
Frequency of occurrence:	If user isn't expiring issues, the log file should be send automatically once pr. month

UC15: Manage Company

Name	Value
Name	 Manage Company
Use Case ID	15
Rank	<Unspecified>
Primary Actors	User

Flow of Events

1.Need for  User to save or edit company's information
2.  User starts manage company
3.System presents company information if available (If information available go to no.4 if not go to no.)
4.  User edit available information (User goes to no.6)
5.  User enters basic company information (cvrNo, name, address ,phone, email, homepage, bankInfo)

6.System saves Company's information
7.  User end manage Company

Extension

2.a. At any time system fails
1.  User start a new recording and retrieve the information.
5.a.  User types in wrong information
1.  User resets fields and try again
6.a. System fails to save
1.  User restarts system
2.System restarts
2.1.System fails to save again.
2.1.1.  User restarts computer and goes to step 1
2.2.System saves company information
7.a. At any time system send error message
1.  User press F1 and are shown Help Menu
2.  User read and tries understanding information in Help Menu
2.1.  User do not understand Help Menu
2.1.1.  User sends email via help desk function
2.2.  User understands and goes to step 3
3.  User tries again

Details

Name	Value
Level	User Goal
Complexity	Medium
Preconditions	Information to be saved is available.
Post-conditions	Company information is saved
Author	Adams
Frequency of occurrence:	approximately twice a year

UC18: Create User defined Spreadsheet

Name	Value
Name	Create User defined Spreadsheet
Use Case ID	18
Rank	Medium
Primary Actors	User

Flow of Events

1.Need for creating user defined spreadsheet
2. User start Create User defined Spreadsheet
3. User enters numbers of columns(noOfColumn)
4. User enters numbers of rows(noOfRow)
5. if named needed on columns User enters name for column (nameOnColumn)
5.1. loop
until no more columns
end if
6.System present spreadsheet
7. User enters data
8.System saves spreadsheet
9. User ends Create User defined Spreadsheet

Extension

8.a. System can't save spreadsheet
1. User restart system
2. User enters data again
3.System saves spreadsheet

Details

Name	Value
Level	User
Complexity	High
Use Case Status	Name Only
Implementation Status	Scheduled
Preconditions	User have data and know how many rows and columns there should be used
Post-conditions	Spreadsheet with data is saved
Author	clausbeck
Frequency of occurrence:	If user isn't expiring issues, the log file should be send automatically once pr. month

UC19:Display ToDo List

Name	Value
Name	 Display ToDo List
Use Case ID	19
Rank	<Unspecified>

Flow of Events

1.Need for order list occurs
2.  User starts generation of the list
3.System shows result (Orders || Null)
4.  User ends  Display ToDo List

Extension

2.a. System fails
1.  User restarts system
2.System starts again
2.1.System fails again
2.1.1.  User restarts computer and goes to step 1
4.a. At any time system send error message:
1.  User press F1 and are shown Help Menu
2.  User read and tries understanding information in Help Menu

2.1.  User do not understand Help Menu
2.1.1.  User sends an email via the helpdesk function
2.2.  User understands and goes to step 3
3.  User tries again

Details

Name	Value
Level	User Goal
Complexity	Medium
Preconditions	User is logged in, Orders are present in the database
Post-conditions	List over orders is presented in the GUI
Frequency of occurrence:	Approximately 1000 times pr year

Appendix C – Mock up and correspondence

Mockup screens && correspondence with KJMC

In this chapter we are showing the full mockup and the correspondence with the company. The following pictures and text are taken from the PDF file, that were send directly.

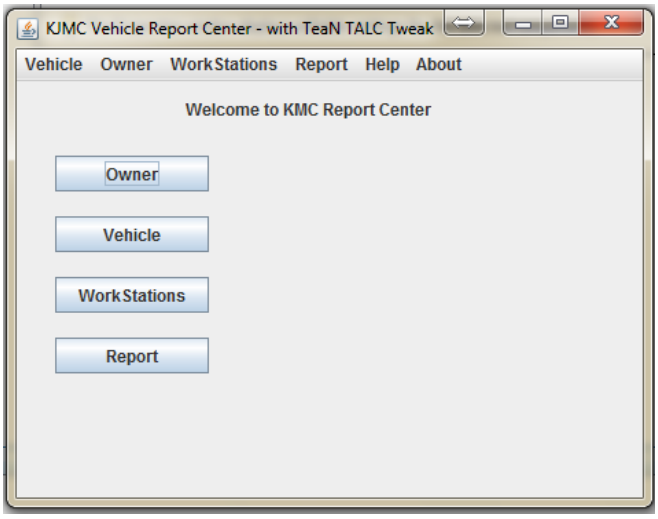
E-mail to KJMC

-----Original Message-----
From: "Claus Pallisgaard Beck" <beck@deoda.dk>
To: "Klaus Klaus" <kni@kjmc.dk>,"wilbers@kjmc.dk" <wilbers@kjmc.dk>
Cc: n.nyvang@hotmail.com,"'LarsSchou'" <lars.sc@gmail.com>,"tvupper@gmail.com" <tvupper@gmail.com>, katngire@hotmail.com
Date: Wed, 06 Oct 2010 11:10:12 +0200
Subject: IdeertilSystemet - kigpådemogkomgernemedin/output

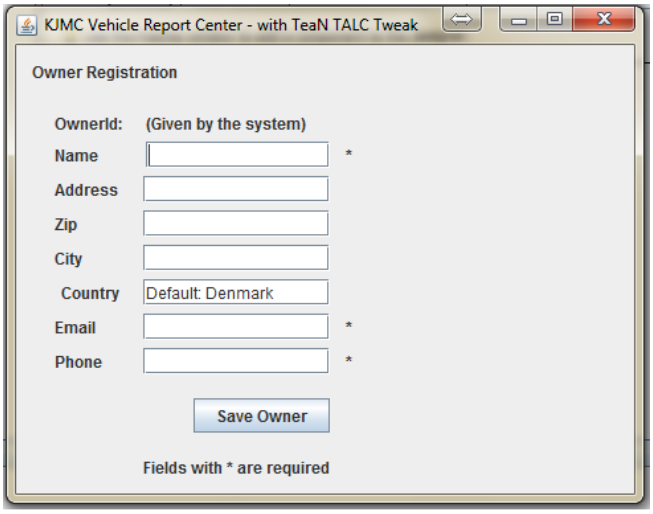
Hej med jer,

Hermed et lille oplæg, dette er IKKE det endelige design, men mere så vi kan få en ide om det er de rigtige ideer vi har, samt om det er de rigtige oplysninger vi samler ind for kunder og køretøjer. Samt det er de rigtige oplysninger vi samler på de forskellige arbejdsstationer.

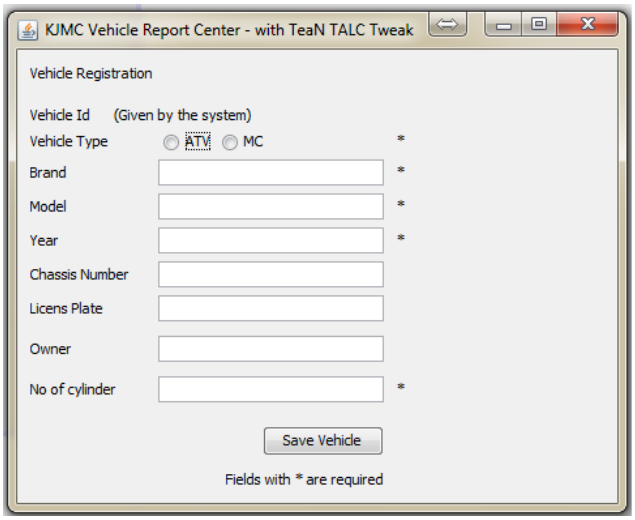
Der er en Velkomst med mulighed for følgende



Ved owner kan indtastet disse oplysninger



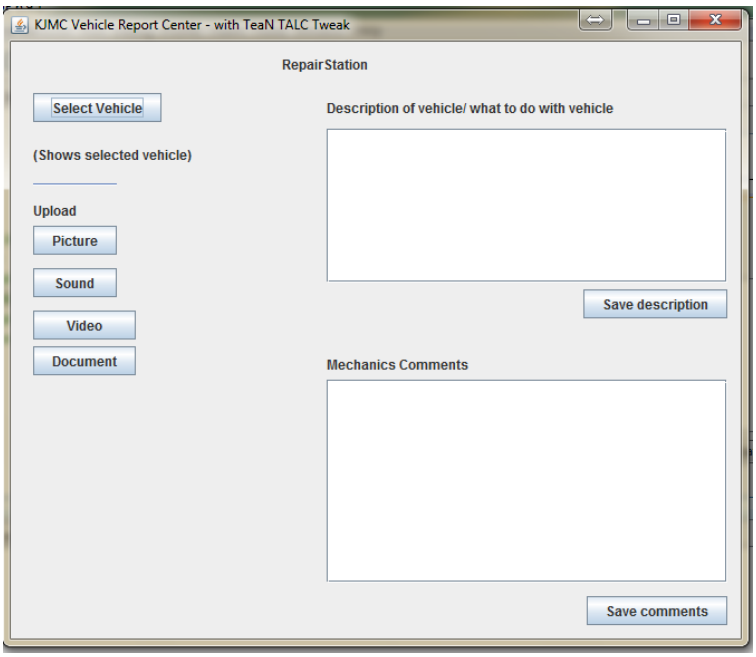
Ved Vehicle kan indtaster disse oplysninger. Indtastning af antal cylinder bruges når vi skal lave et ”whiteboard” i EngineStation



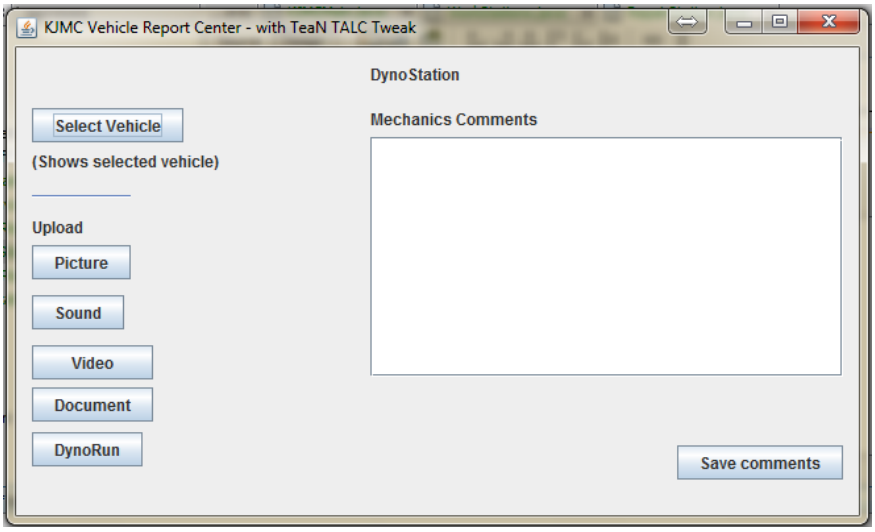
Så ankommer man til repairStation (værkstedet) som er første Workstation.

Her skal vælges et Vehicle, man kan skrive en Description af hvad der skal ske med Vehicle. Mechanic comments er til beskrivelse af hvad der er gjort(man kan også lave en felt til Interne notatet?)

Der kan uploades følgende



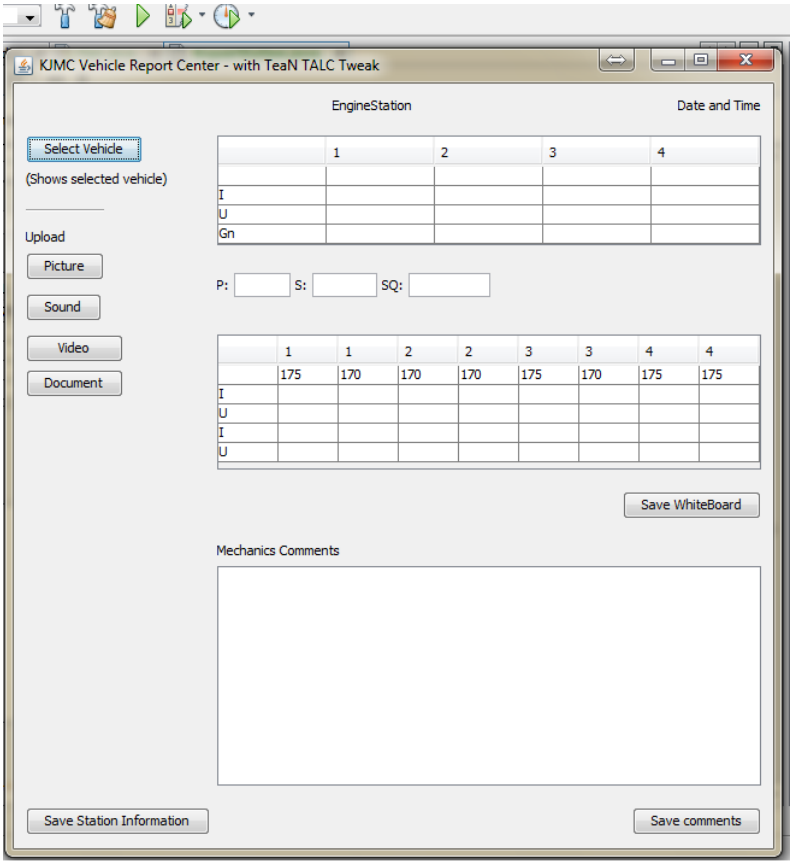
Herefter kommer man til DynoStation, der kan gøres følgende



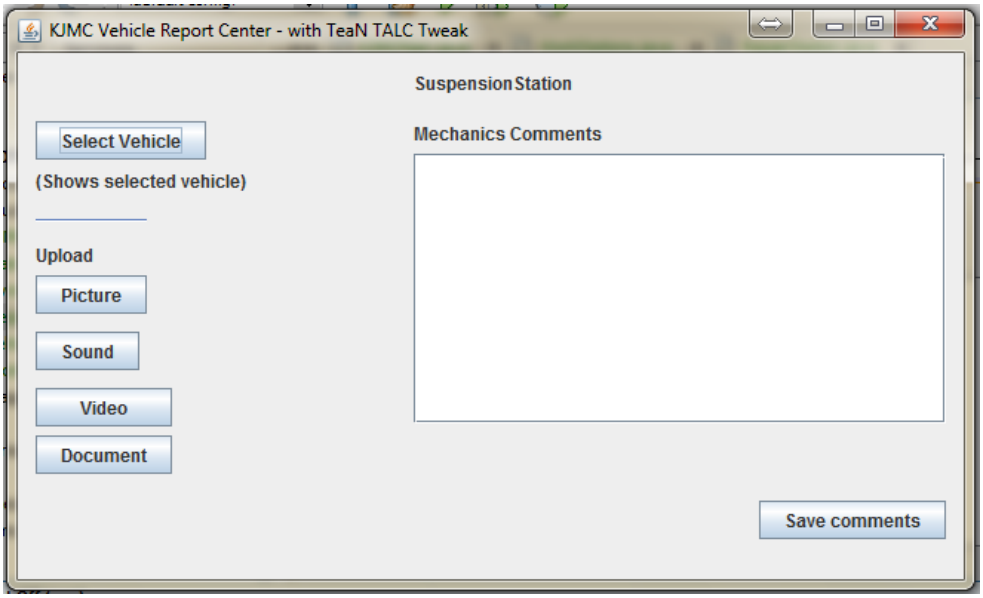
Herefter til EngineStation. Her var det vi skulle bruge nogle oplysninger fra Jer om hvad der skulle kunne registreres. Vi har pt. Sat følgende ind, som vi har fundet ud fra det i fortalte.

”WhiteBoards” oprettes med det antal cylinder der nu er til den valgte Vehicle.

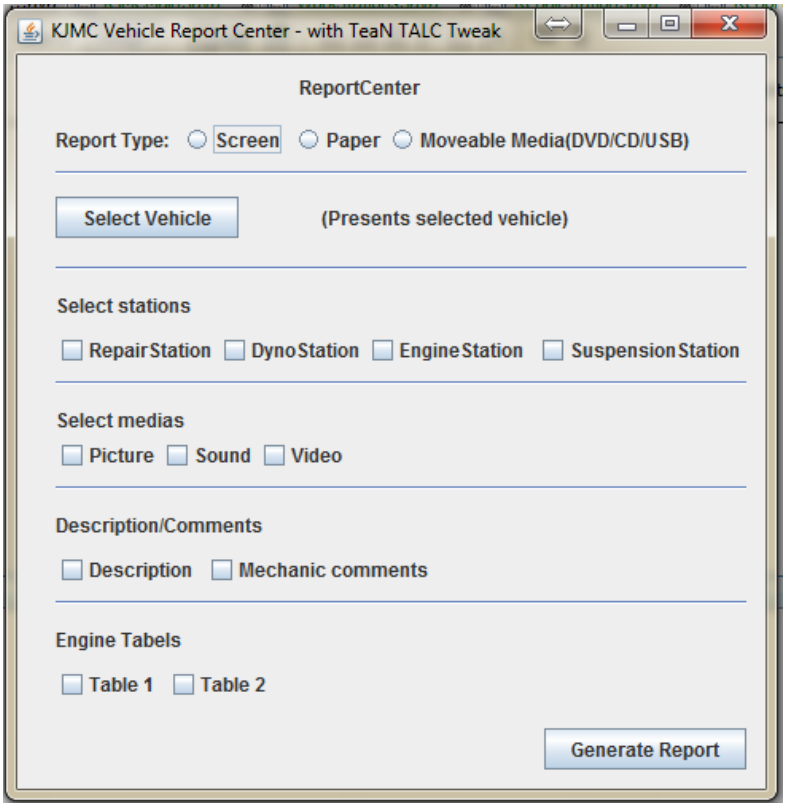
Man kan gemme en version af ”whiteBoardet” hver gang man laver ændringer, det bliver gemt med en dato og tid. Vi vil gerne have at vide om der er mere der skal på. Vi husker noget om nogle (shims?, nogle ”hår” der kunne ligges ind imellem noget, disse er standard mål, så hvis vi kan få oplysningerne på dem, kan vi ligge dem i et rullegardin, så man kan vælge ud fra det mm)



SuspentionStaion kan der gøres følgende



ReportCenter, her kan vælges mellem hvilken ”type” report der kan genereres, samt hvilke oplysninger der skal med på den genererede report.



In og output modtages gerne snarest ☺’

Igen husk dette er IKKE designet men blot en tankerække fra os

Som udgangspunkt arbejder vi mod at kunne lave en screen report (de andre må komme hvis tiden er der). Der er gjort forberedelser til et tidsberegning system, dvs. at man logge ind med et ID og hver gang man vælger et Vehicle i en station, startes et ur, når man lukker stationen igen slukker uret, det samlede resultat skulle gerne komme med et samlet tidsforbrug på den enkelte Vehicle, men igen er det ikke en feature der komme med i denne version med mindre tiden er til det.

Lad høre fra jer, hold kablet stramt og husk at livet først starter i det røde felt(er det ikke sådan???...)

As the company is Danish, we did send the e-mail in this language. When we look in the mirror, we could have done this in english, but actually we never thought about it.

Hilsen
TeaN TALC
(altså os ik...)

KJMC´s 1st reply

-----Original Message-----
From: "Wilbers Servicecenter Danmark" <wilbers@kjmc.dk>
To: "Claus Pallisgaard Beck" <beck@deoda.dk>
Cc: "Klaus Nielsen - KJMC" <kni@kjmc.dk>
Date: Wed, 06 Oct 2010 12:23:41 +0200
Subject: Re: Ideer til Systemet - kig på dem og kom gerne medin/output

Hejsa

Det ser sgu godt ud.....

min rettelse er i engine station og suspension

Engine station skal indeholde flere "Whiteboards", og det ville vøre fedt hvis disse whiteboards var tomme og kunne importeres efter behov og indsættes, dvs. at jeg kan "hente" en med x antal celler vandret og y antal celler lodret, og ændre efter behov.... og selv indsætte både det jeg måler og parameterene som måles.....

Det samme gør sig gældene i suspension..... tomme charts der kan bruges efter behov, importeres og lægges ind, som et regneark hvor jeg kan sætte flere rækker eller søjler ind efter behov.....

Det fremtidsikre også programellet jeg kender ikke mit behov om flere år.....

Med venlig hilsen

Mikael

Wilbers Servicecenter Danmark
Tlf.: +45 4494 1935
Mail: wilbers@kjmc.dk
www.kjmotorcykler.dk

KJMC´s 2nd reply

-----Original Message-----
From: "Klaus Nielsen - KJMC" <kni@kjmc.dk>
To: "Claus Pallisgaard Beck" <beck@deoda.dk>
Date: Thu, 07 Oct 2010 01:30:56 +0200
Subject: Re: Ideer til Systemet - kig på dem og kom gerne medin/output

Nice.....
udover Michaels tilføjelser har jeg flg :

Vi vil gerne kunne trække Matties servicedokumenter ind, enten direkte eller som pdf - muligt ?

Under "Vehicle" ville det være fedt, hvis den kunne udvides med VO (sender dig kopi af den gamle, vi bruger idag). Så kunne vi bruge systemet til VO, og så bygge videre med det, kunden skal have lavet i løbet af vinteren

---

Har du set sidste nyt fra KJMC RR, læs mere her

Vi har fået RAM AIR på vores rullefelt, læs mere her

Med venlig hilsen

Klaus Nielsen
+45 4494 1935
kjmotorcykler.dk

Appendix D - Revised timetables

Appendix E – Business Analysis

TEXT NEEDED HERE

2.1 Purpose
In this part we need to analyze the critical non-functional requirements, and set the possible objectives at the end objectives mile stone will be achieved. After doing business analysis we have decided to deal with elaboration and trying to impermanent some use cases like registering vehicle and others

2.2 Activities:
The following will be done in this iteration.

Business model: This gives the over view of this ***KJ Motorcycle*** Company. From here we are going to implement what the company requires from us us.

Vision: this shows how we see the company operating the system at the end when we have finished.
Eristic stakeholder's requests: here we are finding out who is interested in the system and what he needs the system to do.
Find actor and use case: here we specify actors for the system and identify the necessary use case.
Detail use Case: Find an important use case and present it in a fully dressed format
Glossary: a list of words used in the project.
Do project management conclusion and plan the next step according to the project plan.

2.2.1 Business model: this is helping us to analyze the environment of ***KJ Motorcycle*** Company we are dealing with in our project.

2.2.1.1 Context:
KJ Motorcycle company is located in Herlev and this is outside Copenhagen, Denmark's capital city. But it operates in
almost in the whole of Denmark, including Copenhagen areas, and Jylland as well but it has customers from other Nordic countries mostly Norway
The Company is quite young as it has only operated for some few years, now the company has about 3 workers. It provides quite expensive services, and this means it serves quite rich individuals.

Five Forces Analysis

Five forces analysis is important in business analysis as well, in a way that it helps in contrasting a competitive environment, and it looks at five key areas (the threat of new entry, the power of buyers, the power of suppliers, the threat of substitute, and competitive rivalry.

MARKETLEVEL

- **Customers** and competitors (customers -> offering) **-** ***(Porter 5 forces)***

  There are 3 kind of customers

  1. Buying from the internet (parts for MC and drivers)
  2. Buying in shop, buying parts for MC and drivers, buying service or adjustment on MC
  3. Buying rebuilds (big or small) of MC (motor, suspension etc.)

  Customer 1: The main part is from DK, most of them from outside of Great Copenhagen area. There are a small part from other Nordic country's mostly Norway

  Customer 2: From DK and mainly from Sea land

  Customer 3: From all over DK

  Common for all customers are that they drive MC and want some extra parts or styling for their MC.

  Special for customer 3 is that they are enthusiast (or nerds), and want something special.

  MC customers normally want to use the same company for everything (if they are satisfied. That means : parts, service, winter storage, buy and sell) Therefore the company most fit these demands to keep customer.

- Customers and **competitors**

  Main competitors if we talk about parts and styling is other web shops, mainly from outside of DK (they have lower VAT, also cheaper prices (DK is a very small MC country, therefore DK importers can make big orders and get low retail prices)
  Known competitors could be:
  In DK:
  http://www.eracing.dk

  Outside DK:
  http://www.carpimoto.com
  Competitors to the normal workshop (for service etc.) is big, there many MC workshops all over DK, but again MC customers are very reliable if they like the shop/workshop

  Competitors in fully rebuilding MC are nearly non existing, but again so are the customers.

- **Offering - *(Generic Strategies)***
  They have a web shop and a "real" shop for:
  Parts and styling, to driver and MC. Race bike and ATV

  They have workshop for:
  Insurance claims, service, performance optimizing, mounting accessories /styling and winter storage.

  And a workshop for:
  Complete rebuilds of MC

  They offer full packets that means: lease or buy a MC with service (also including tires) and winter storage.

- **Activities and organization - *(Value Chan)***
  Primary activities are:
  Workshop and web shop, leasing
  Shipping for web shop customers and service/rebuilds are done in Herlev
  all leasing and accounting is done from the office in Viborg
  They use GLS for delivering and pickup of goods.

  They are importing some products to Denmark and sell them to other dealers and customers, main brands are:
  Termignoni – Exhaust
  HEL – Brakelines
  DynaPro – Dynometers
  ValterMoto – Stylingparts
  They have many suppliers depending of which kind of products, main suppliers are

Matthies
Bridgestone

- **Resources** (Resources and Competencies) - ***(SWOT)***

  They have mechanics doing all workshop jobs.
  They have a needs of special knowledge (human resource and IT) then they are doing rebuilds(Human) and optimizing(Dynometer – IT resource)

- **Factor Market Suppliers** ( )

  They have to be sure that were products are good, when we talk rebuilds of MC. They must have reliable suppliers.
  When we talking styling parts to MC or drivers there are 2 categories, Lookalike there is cheap and then Official brands there are expensive.
  Normally the Official brands have their products for about 6 months, before they are imitated and sold to a cheaper price, mainly imitated parts come from Hong Kong and China.
  There are several big importers in Europe, getting their own brands done (they are mostly looking like the Official Brands) and selling to a cheaper price, mainly there quality is very good.
  Labor for workshop is not hard to get at this time, but skill labor for rebuilds are more difficult to find. But DK is a small country then you talk motorbikes and everybody knows each other, so you help out.
  For the leasingpart of the company you have one or two main suppliers for capital (banks) and the item you lease out comes from many different suppliers, mainly from Germany.

POTER'S GENERIC STRATEGIES DIAGRAM

Figure 1: Porter's Generic Strategies

The source of competitive advantage for this company is differentiation. It offers special service for its customers and it's too loyal to them as well. Cost leadership in this company is shown when the company try to minimize the costs as much as possible for instance it only have 3 workers here it is trying to minimize human resource costs. Differentiation is shown when this takes an extra mile by advising its customers on their motor vehicles.

Appendix F – Operation Contracts

OC2:enterOwnerDetails

Operation Contract - Register Owner

TeaN TALC - Tweak[MC]

OC2 : Enter owner details(firstname, lastname, address, address2, zip, city, country, email, phone)

CrossRef: Register Owner

Pre Conditions:

A Owner catalog exist.

Post conditions:

- Owner instance o was created.
- o was associated with OwnerCatalog.
- o.ownerId was set by system
- o.firstname was set to firstName
- o.lastname was set to lastName
- o.address was set to address
- o.address2 was set to address2
- o.zip was set to zip
- o.city was set to city
- o.country was set to country
- o.email was set to email
- o.phone was set to phone

OC3:enterWhiteBoardDetails

Operation Contract - Create WhiteBoard

TeaN TALC – Motorbike application

OC3: enterWhiteBoardDetails(whiteboardType)

CrossRef: Create WhiteBoard

Pre Conditions:
Connection to database is established
Vehicle and workstation is selected.

Post conditions:
- Whiteboard Instance wb was created.
- wb.tableName was set to tableName
- wb.rows was set to rows
- wb.column was set to column

OC4:enterWhiteBoardData

Operation Contract - ManageWhiteBoard

TeaN TALC – Motorbike application

OC4 :enterWhiteBoardData(data)

CrossRef: ManageWhiteBoard

Pre Conditions:
Connection to database is established
A Vehicle is selected
Whiteboard is created

Post conditions:
- data was entered
- data was set to data
- data was saved

OC5:uploadMediaDetails

Operation Contracts Upload Media

TeaN TALC – Motorbike application

OC5 :uploadMediaDetails(filename, mediaType)

CrossRef: Upload Media
Pre Conditions:

A Vehicle is selected.
Media is present.

Post conditions:
- media type m was selected
- m was attached to vehicle

OC6:manageOwner

Operation Contracts Manage Owner

TeaN TALC – Motorbike application

OC6 :manageOwner(edit | | delete)

CrossRef: Manage Owner
Pre Conditions:
Owner is selected

Post conditions:
- owner o was instantiated
- o was deleted or o was updated

OC7:enterUserInitials

Operation Contracts Register User

TeaN TALC – Motorbike application

OC6 :enterUserInitials(initials)

CrossRef: Register User

Pre Conditions:

A connection to database is established

Post conditions:

- user initials was set to initials
- initials was set to row on userTable

OC8:enterOrderInformation

Operation Contracts Register Order

TeaN TALC – Motorbike application

OC6 : enterOrderInformation(orderType, descriptionOrder, descriptionSparepart, startDate, expEndDate, expPrice)

CrossRef: Register Order

Pre Conditions:

A connection to database is established

An orderCatalog exist

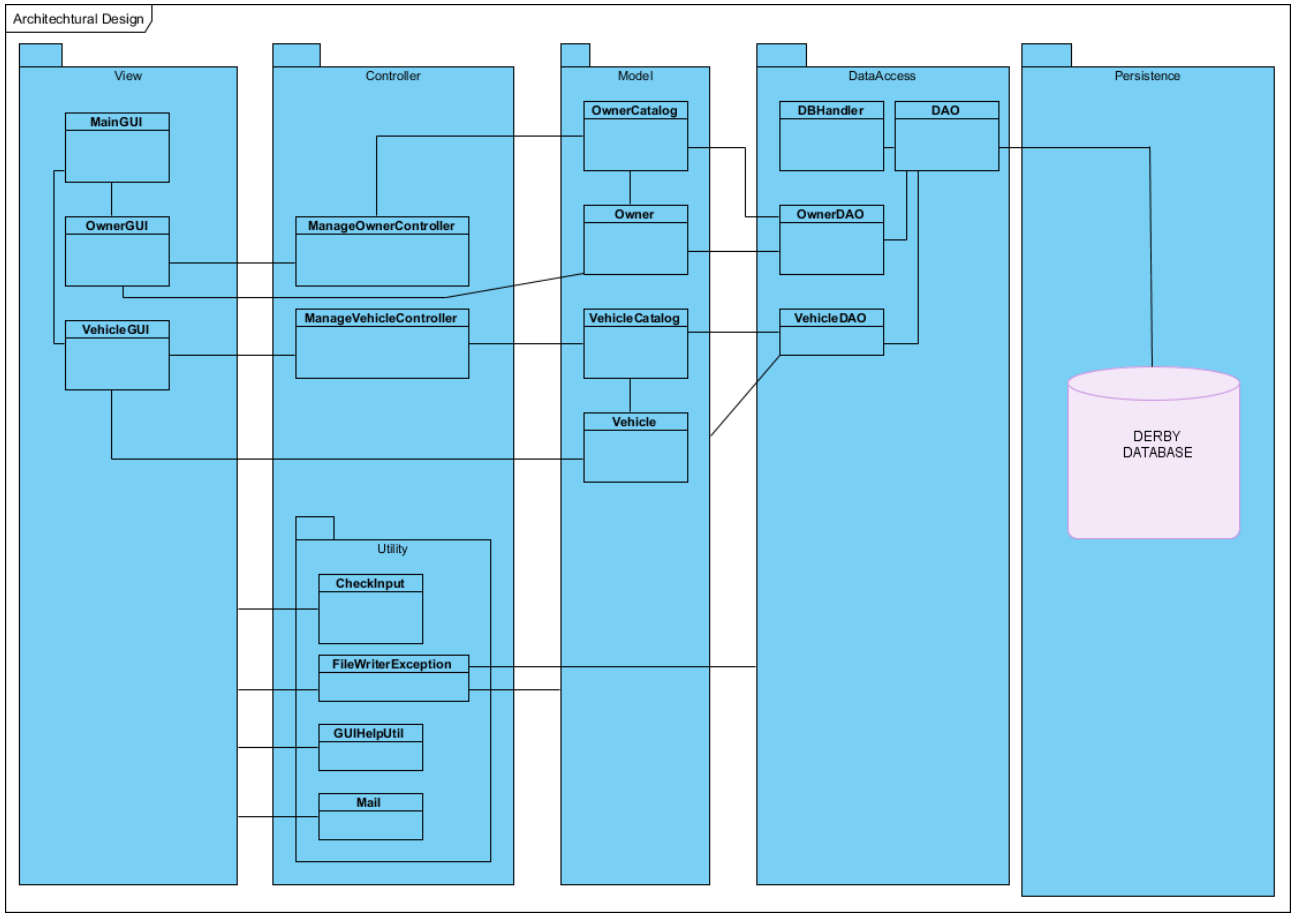
Post conditions:

- An order o instance was created
- o was associated with orderCatalog
- o.orderId was set by system
- o.orderType was set to orderType
- o.descriptionOrder was set to descriptionOrder
- o.descriptionSparepart was set to sparepart
- o.startDate was set to startDate
- o.expEndDate was set to expEndDate
- o.expPrice was set to expPrice

Appendix G – Architecural analysis/design

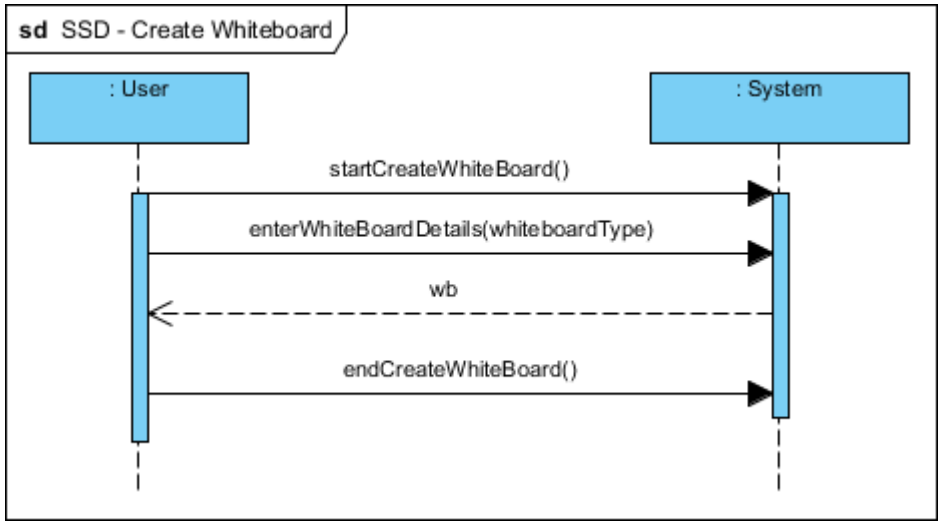
Architectural design and impact analysis

Construction 1. Iteration

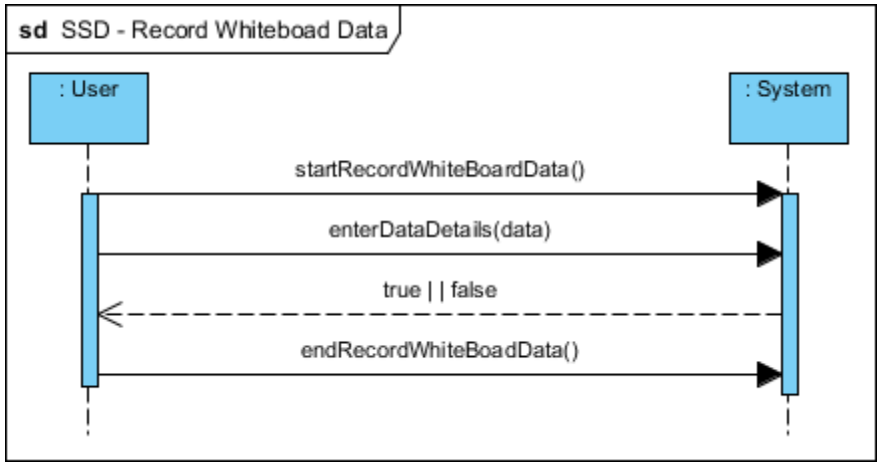


Appendix H – SSD

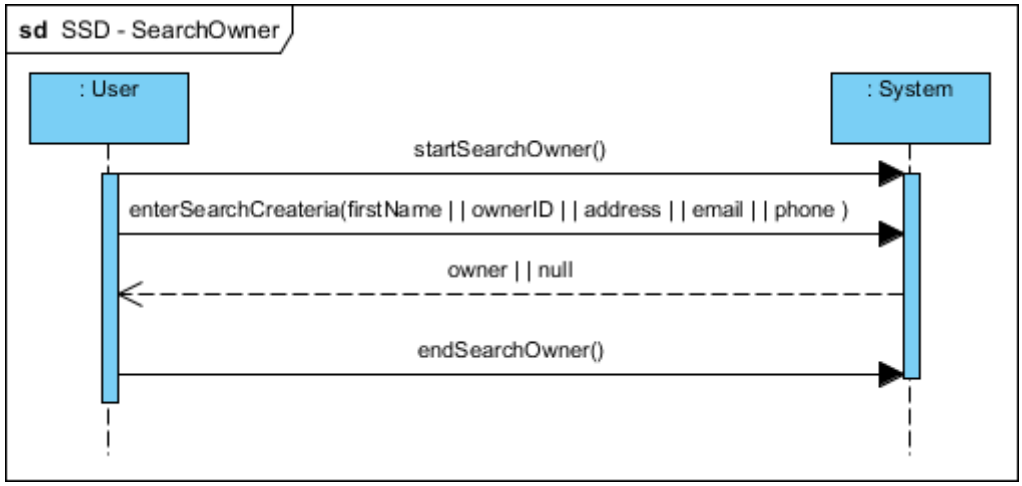
SSD – Create Whiteboard



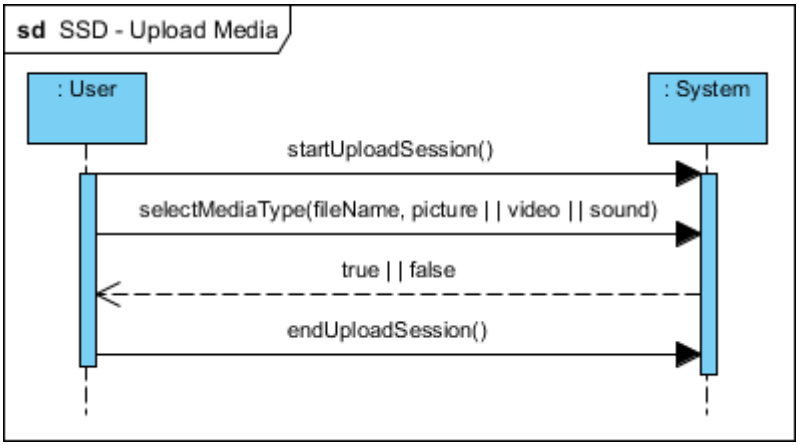
SSD – Record WhiteBoard



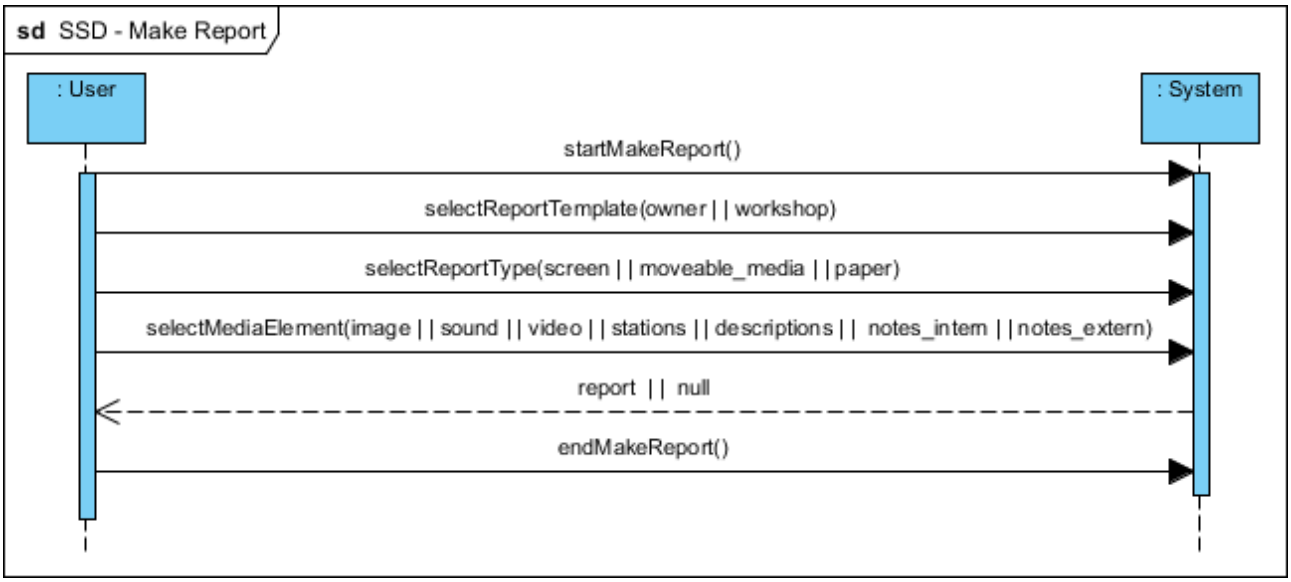
SSD – Search Owner



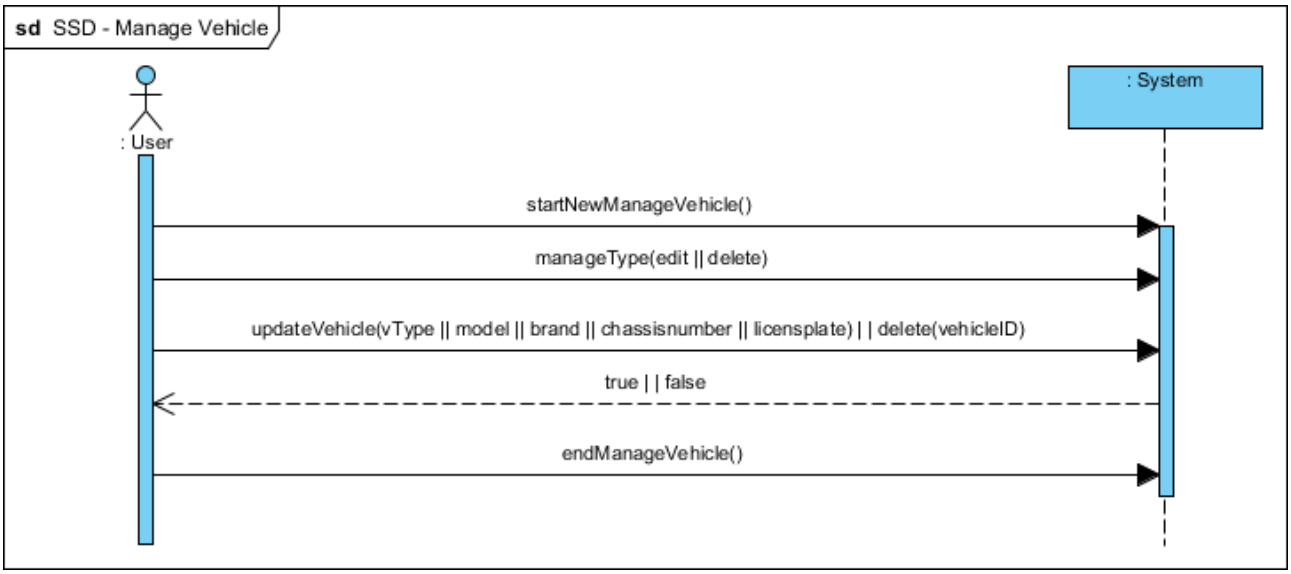
SSD – Upload Media



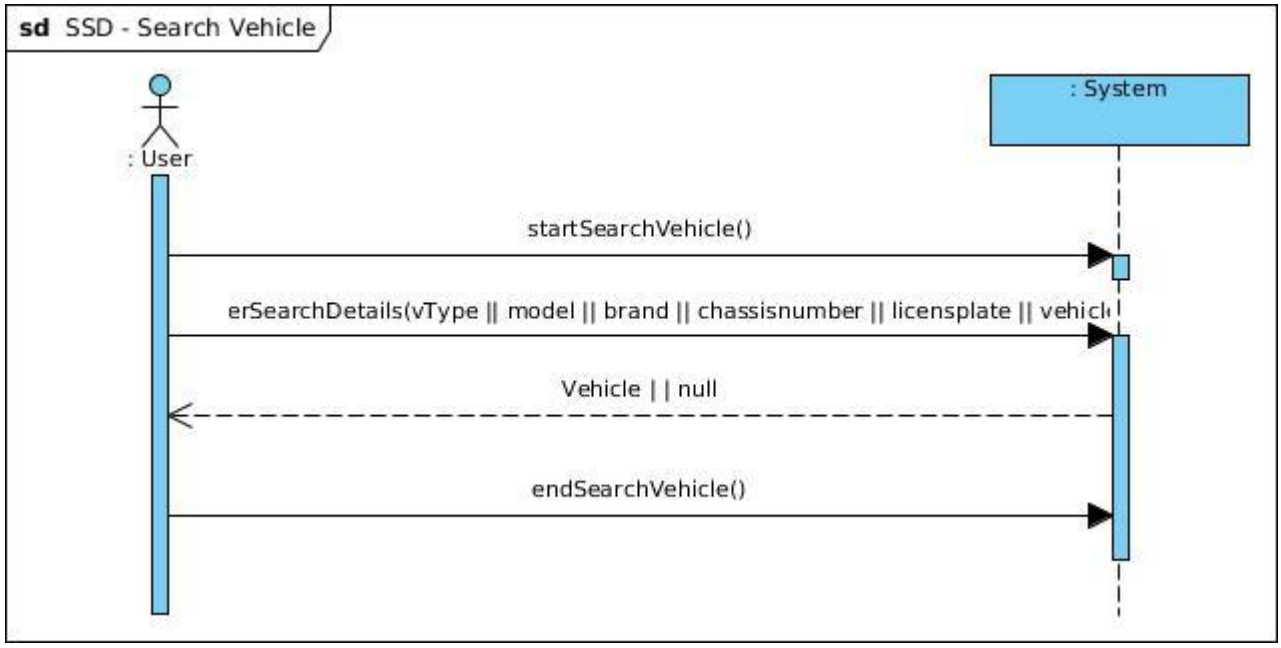
SSD – Make Report



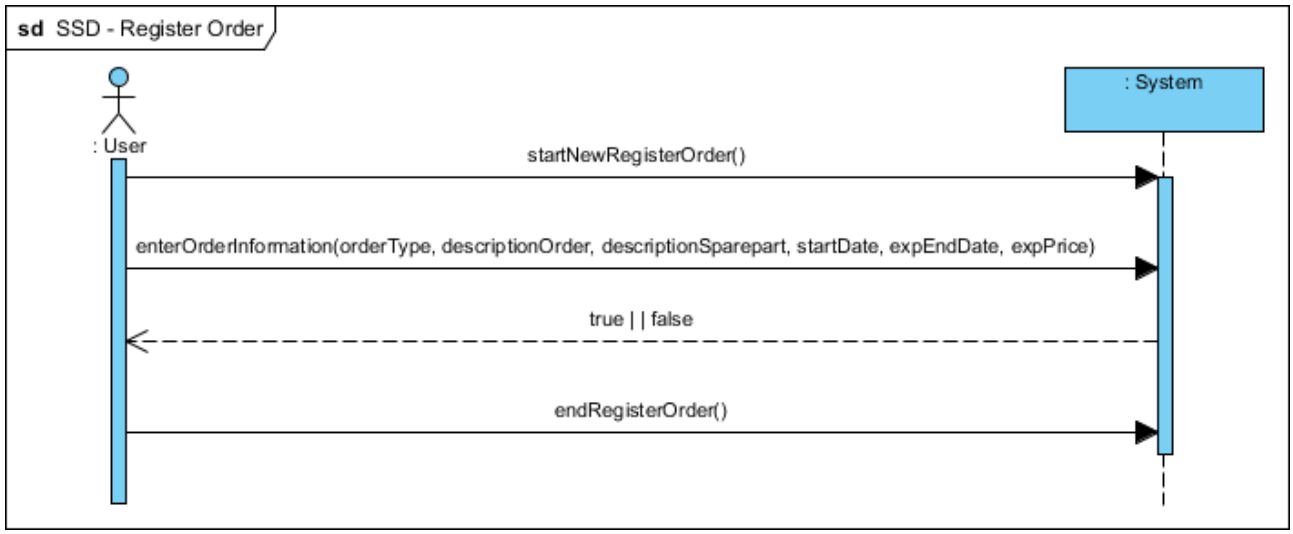
SSD – Manage Vehicle



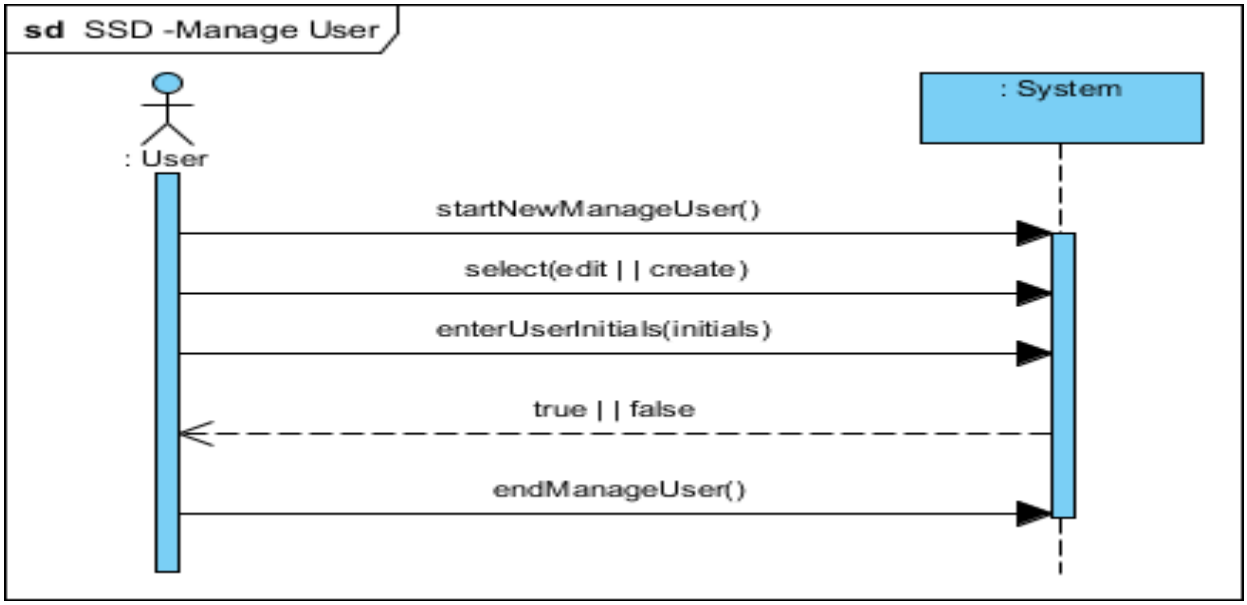
SSD – Search Vehicle



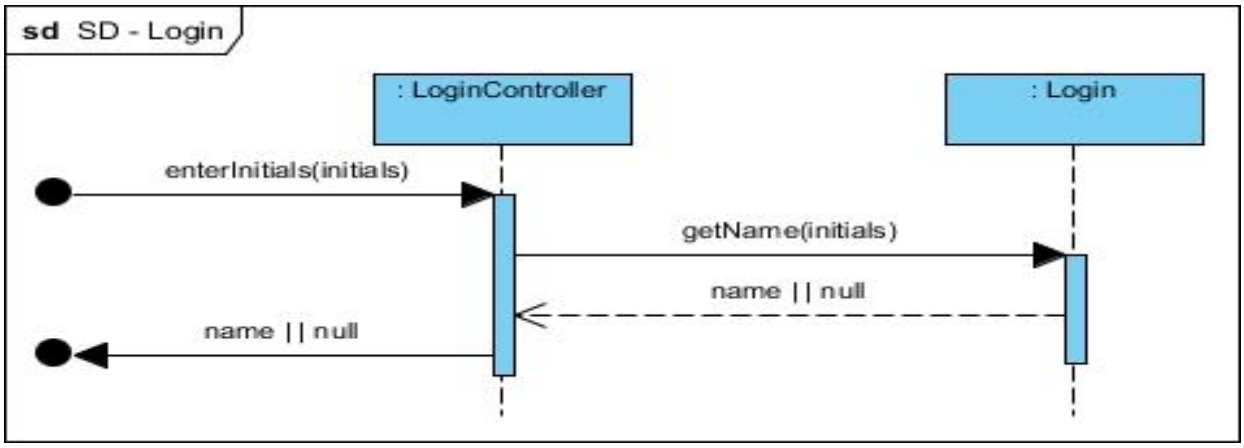
SSD – Register Order



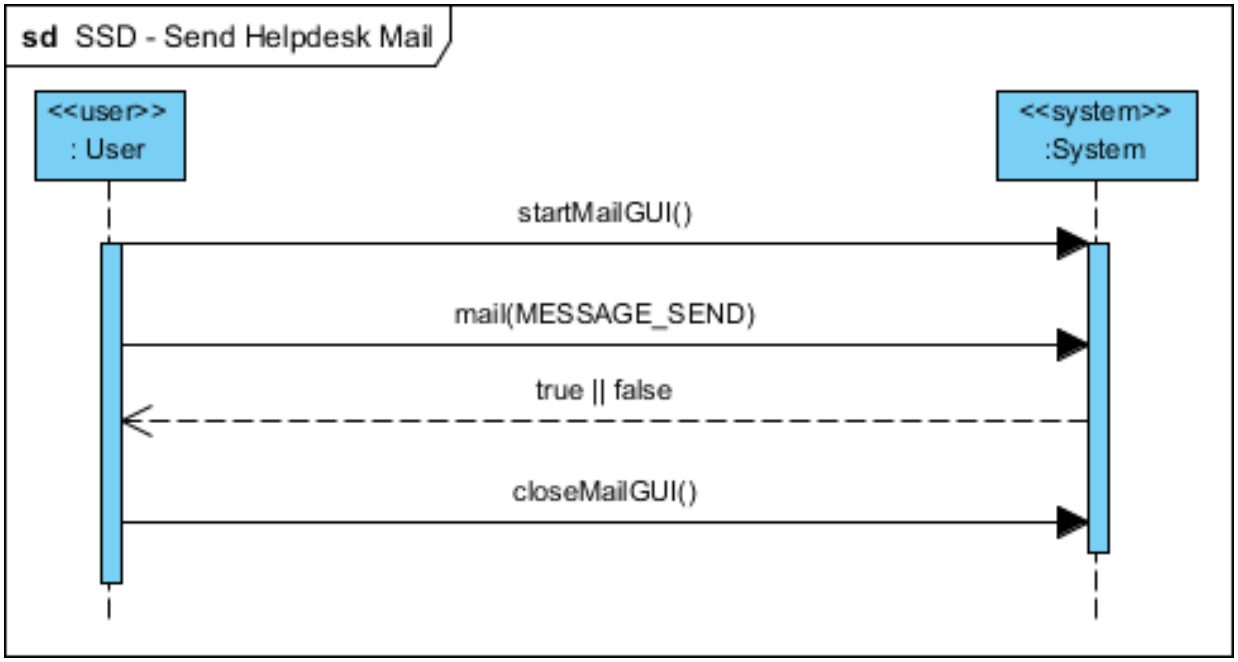
SSD – Manage User



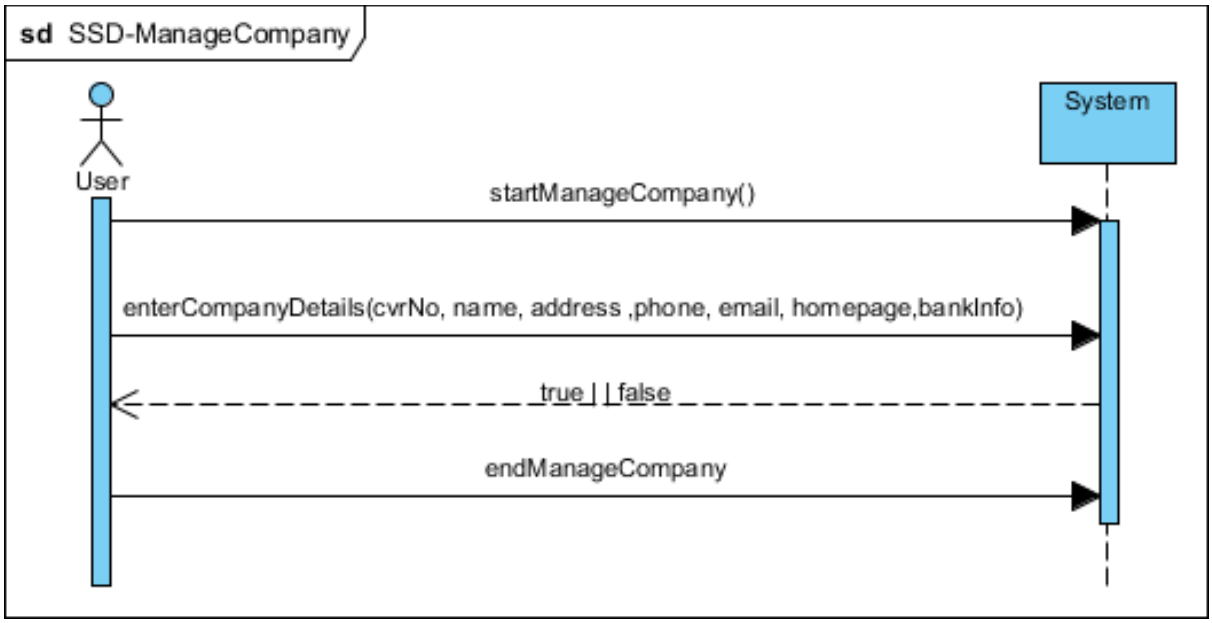
SSD – Login



SSD – Send Helpdesk Email



SSD – Manage Company

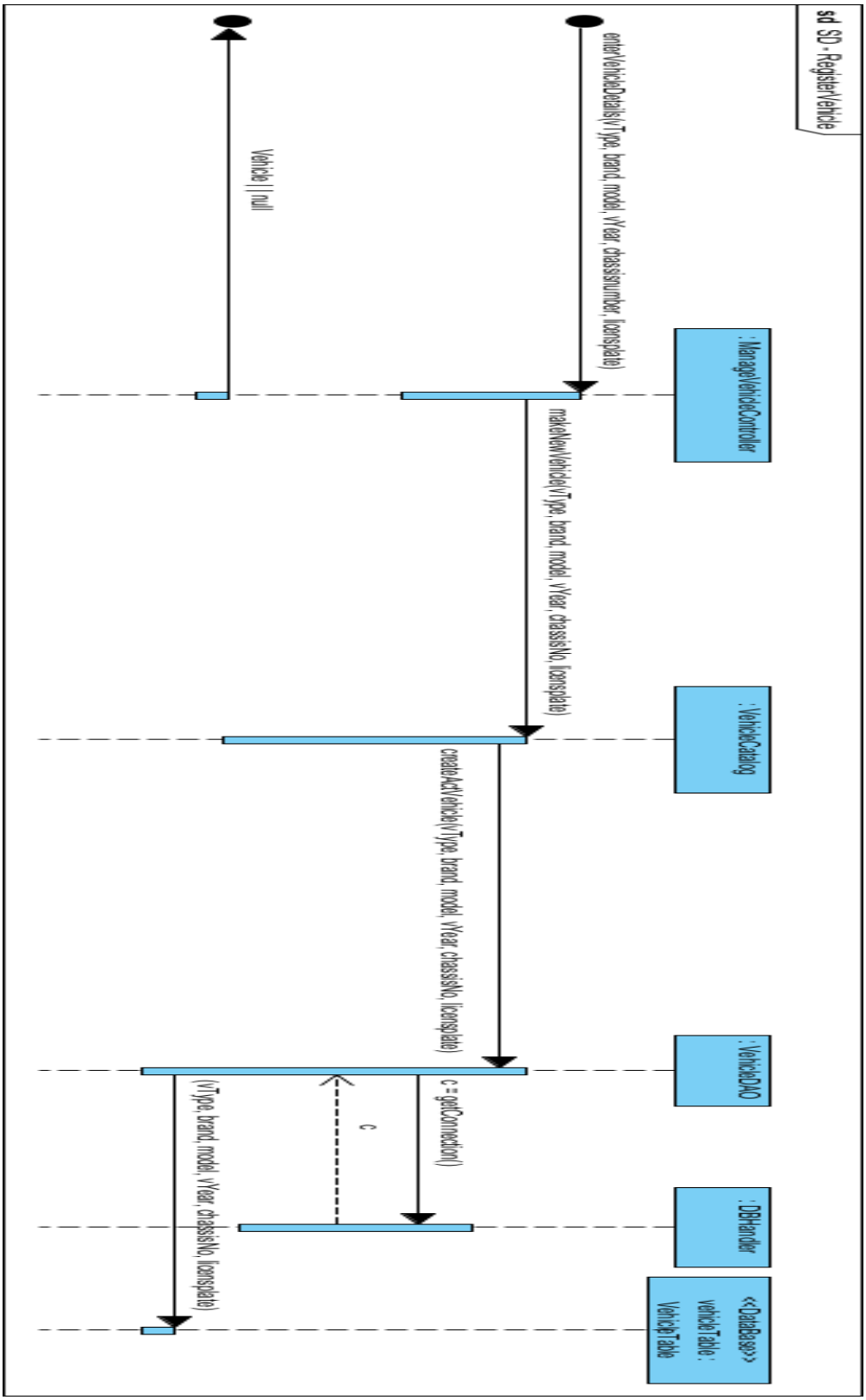


SD – Register Vehicle

This start with the user sending a message to ManageVehicleController with parameters (controller pattern). The controller now sends the parameters further to the VehicleCatalog. The vehicleCatalog now sends the parameters furher to the VehicleDAO – wich calls DBHandler for getting a conection to the database. When the connection is established the vehicle sends the parameters to the Database

Appendix I – SD

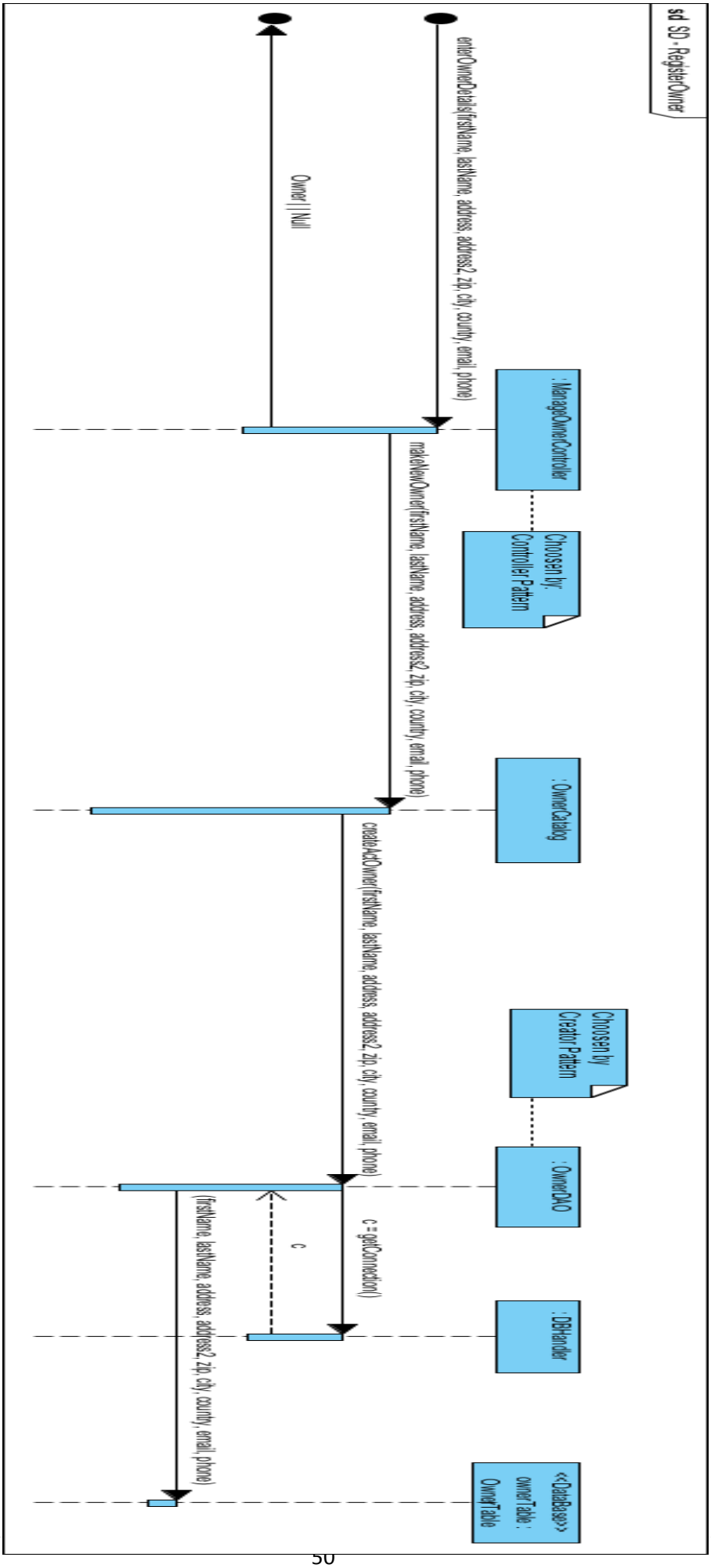
In this appendix there will be a full size diagram, to make the view easier.



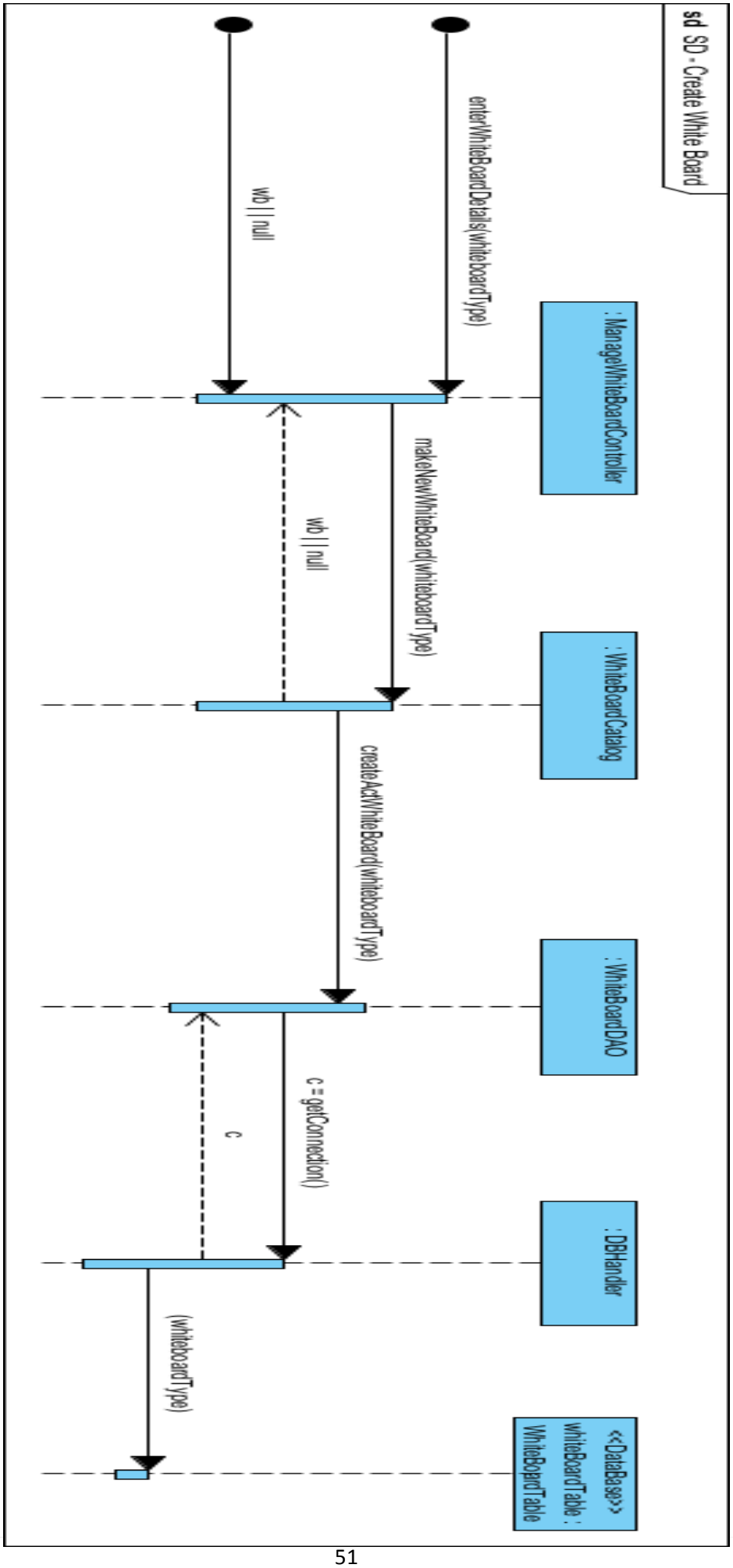
SD – Register Vehicle

This start with the user sending a message to ManageVehicleController with parameters (controller pattern). The controller now sends the parameters further to the VehicleCatalog. The vehicleCatalog now sends the parameters furher to the VehicleDAO – which calls DBHandler for getting a connection to the database. When the connection is established the vehicle sends the parameters to the Database

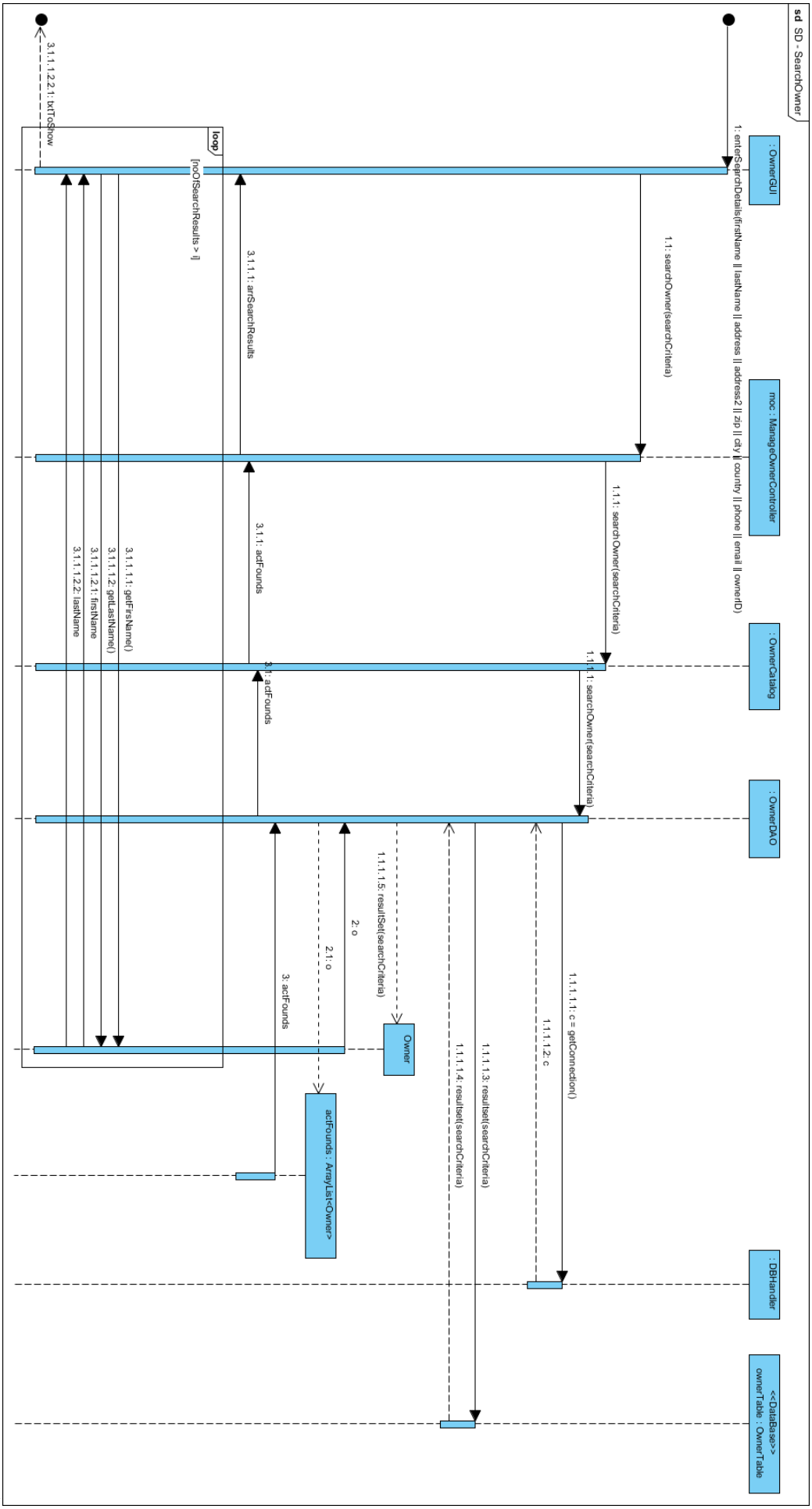
SD – Register Owner



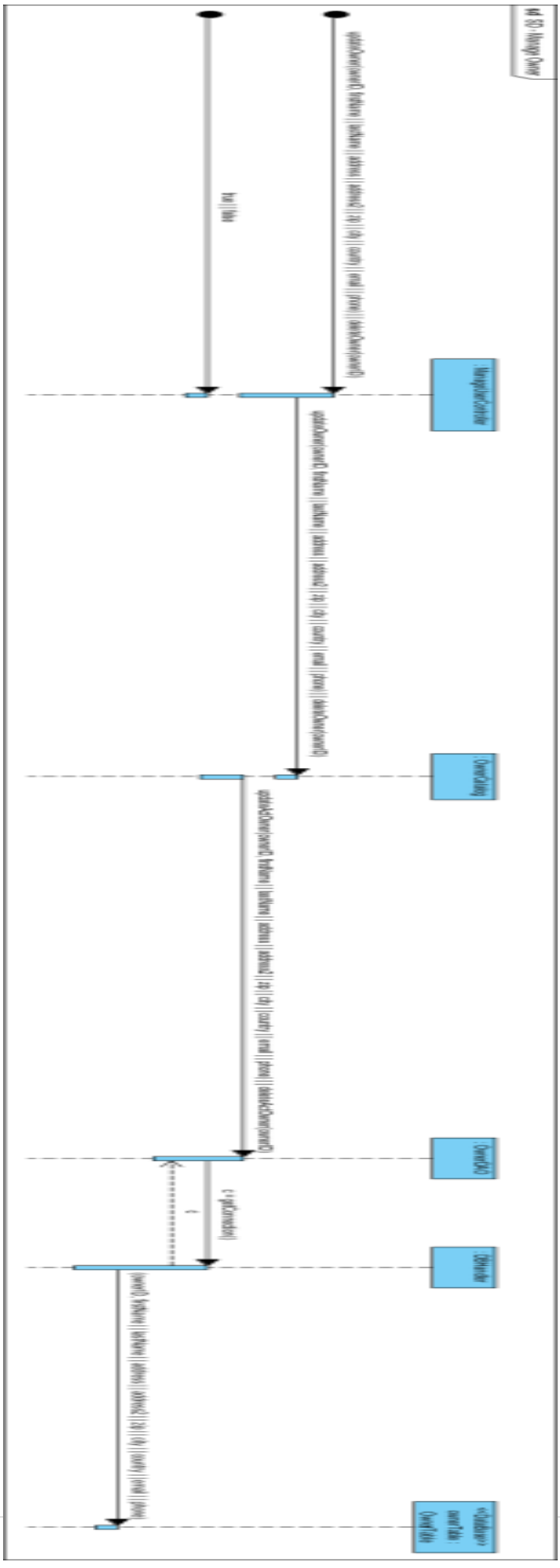
SD – Create Whiteboard



SD – Search Owner

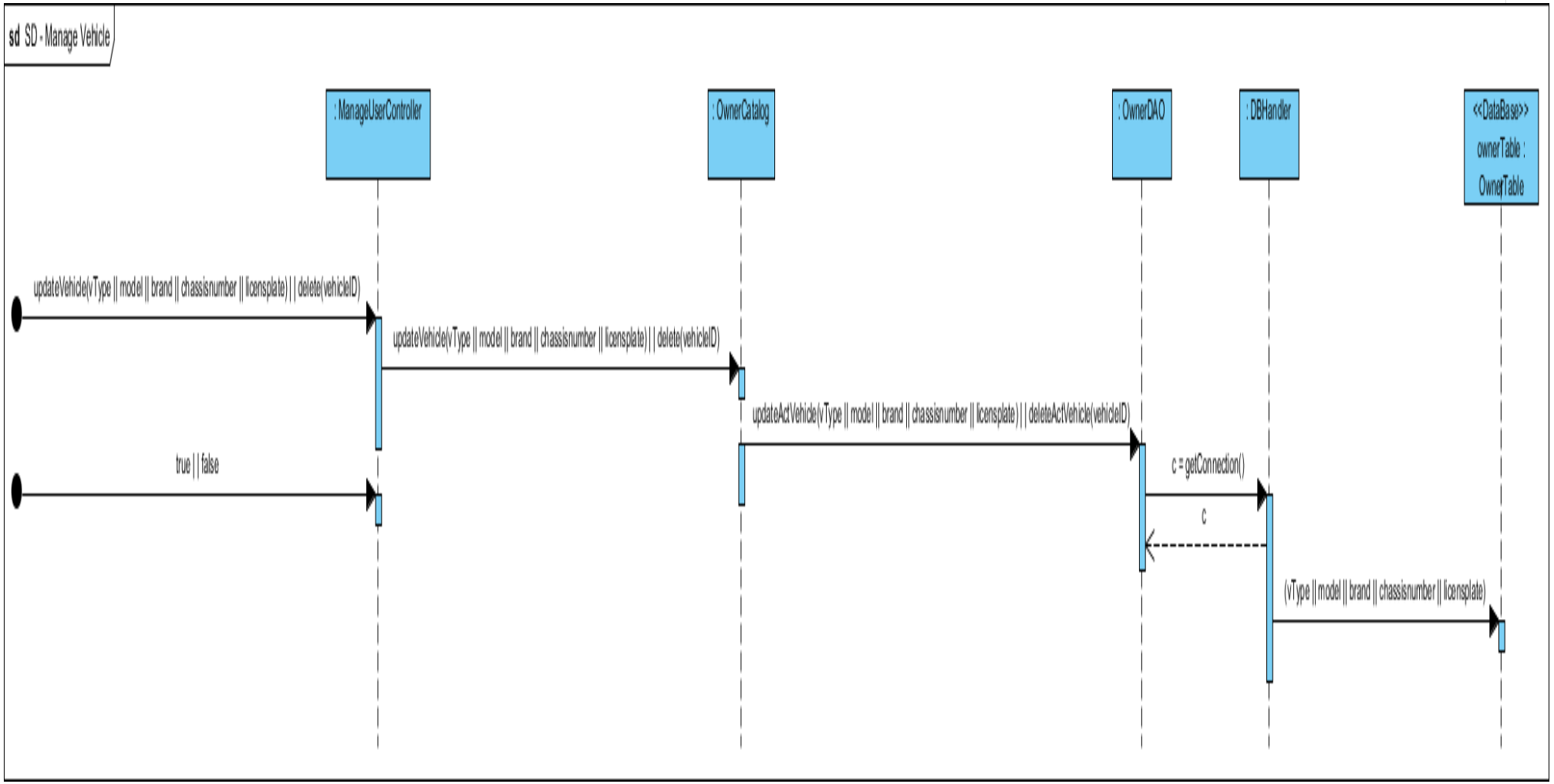


SD – Manage Owner



SD – Manage Vehicle

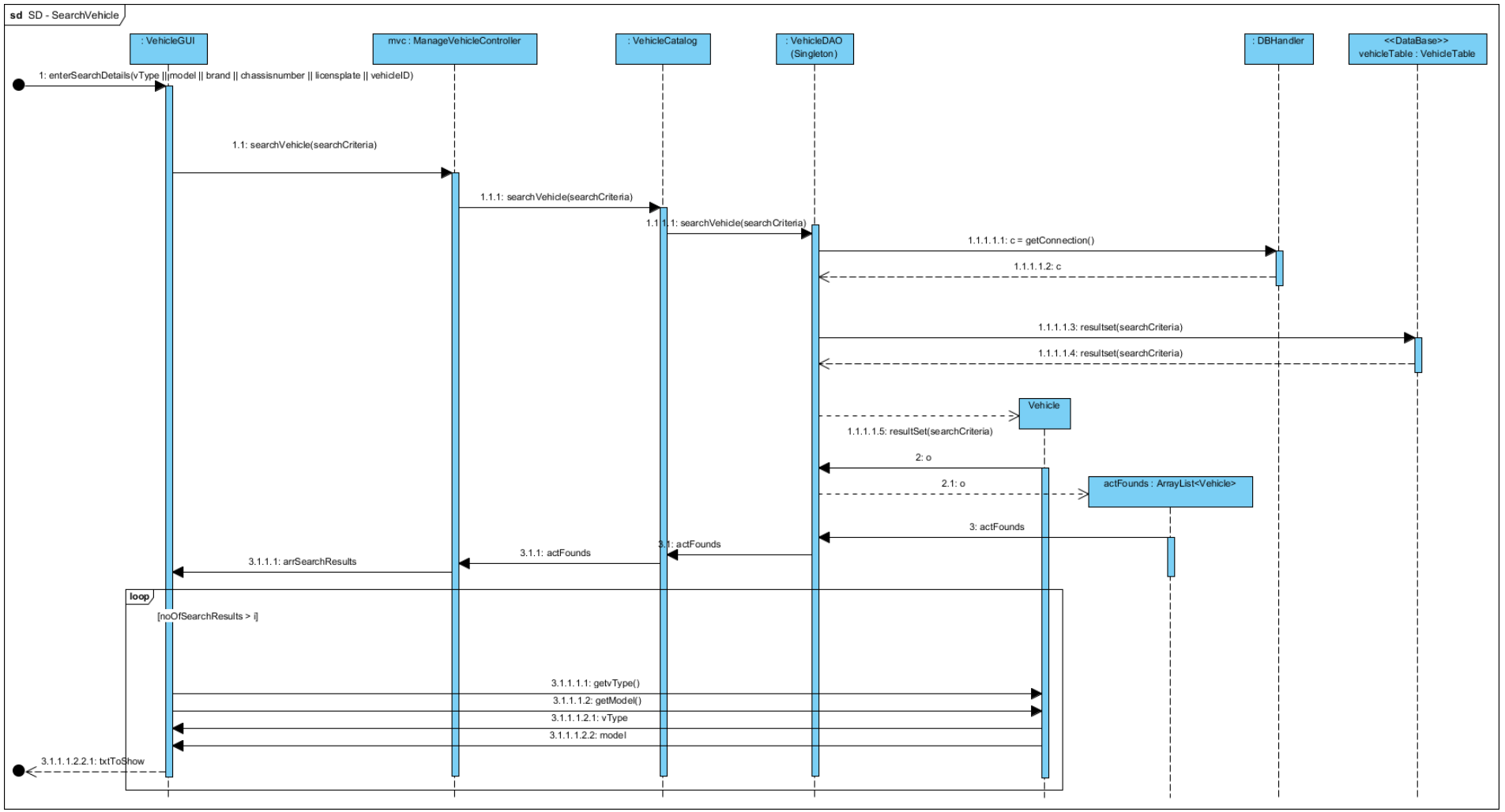
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin placerat, enim vel cursus bibendum, tortor quam sodales elit, et vehicula nisl velit id mauris. Nam eu mollis odio. Nunc id mi in arcu viverra molestie. Nam at nibh eu nibh fringilla cursus. Morbi eget enim quis nisl pellentesque aliquam. Aliquam erat volutpat. Pellentesque consequat mollis erat, vitae euismod diam ultrices ac. Cras interdum suscipit nisi ut ornare. Sed feugiat elit quis neque feugiat accumsan. Suspendisse potenti. Donec et pulvinar sapien. Cras dapibus ipsum quis elit consectetur venenatis. Integer dignissim lobortis tempor. Praesent in mi felis. Sed ipsum ante, scelerisque in viverra sed, viverra eu enim.



SD – Manage Vehicle

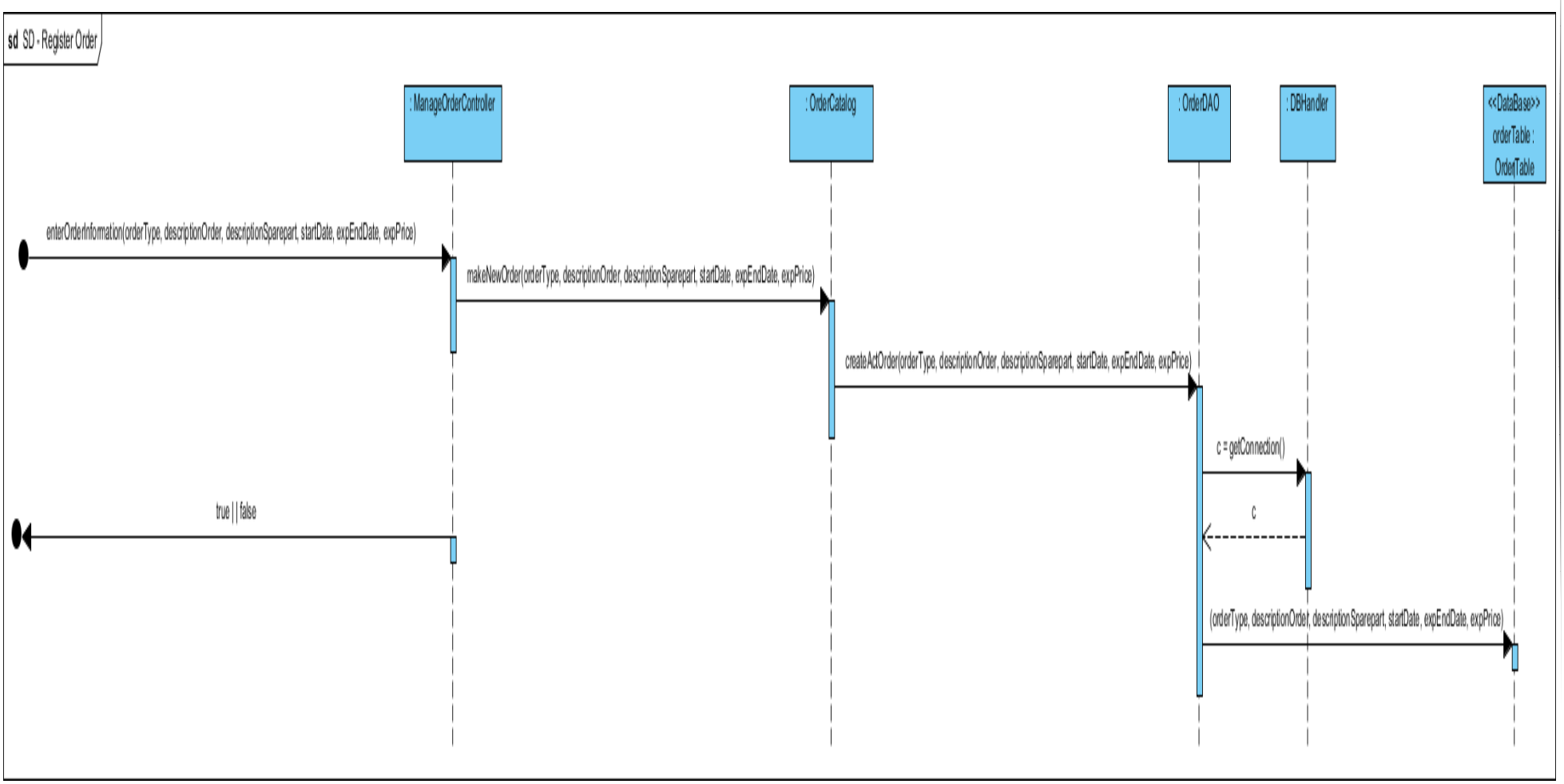
SD – Search Vehicle.

This starts with the user sending searchCriteria to the ManageVehicleController. The ManageVehicleController now sends the message further to the VehicleCatalog, witch sends it further to the VehicleDAO. The vehicleDAO now ask the DBHandler for a connection to the Database. When the connection established the database return a resultset with the found vehicels. All the results are now packed as objects in a list in the VehicleDAO and return to the GUI. The gui now pack them out for presentation



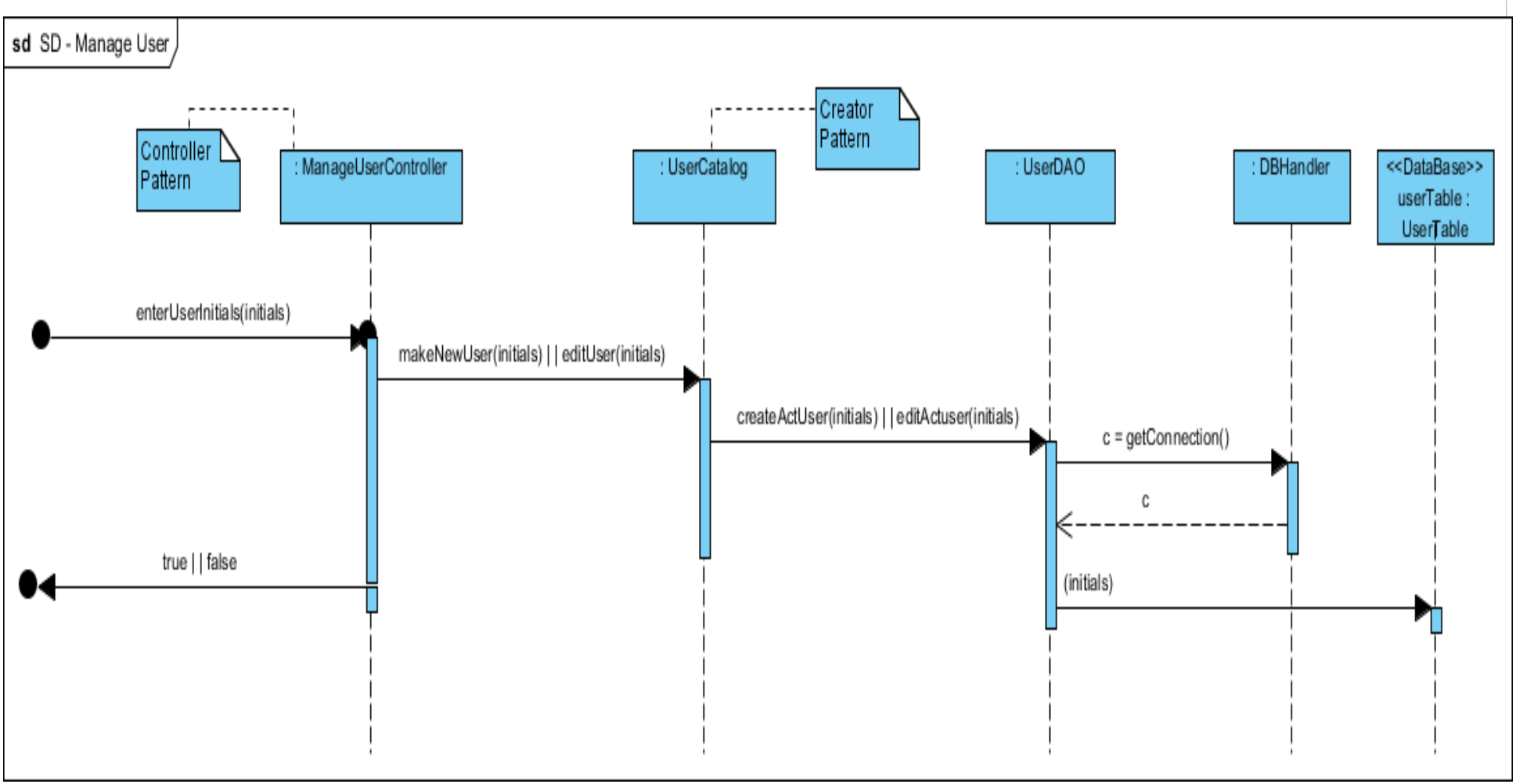
SD – Register Order

This start with the user sending a message to the ManageOrderController with all the parameters necessary to make a new order. The ManageOrderController now send the parameters to the OrderCatalog. The orderDAO now sends the parameters to the OrderDAO, witch calls the DBHandler for a connection to the Databse. When then connection is established the OrderDAO sends the parameters to the Database.



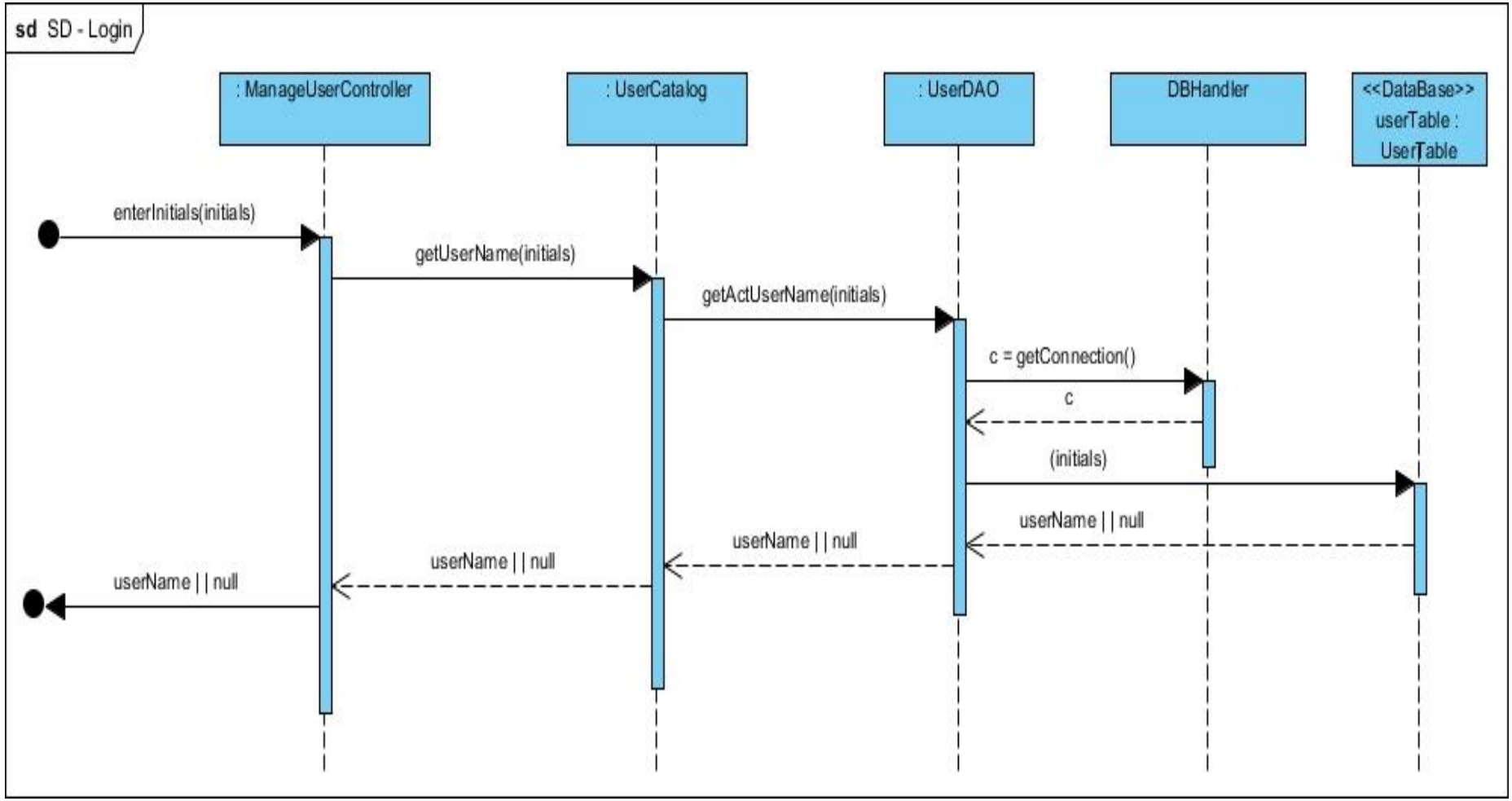
SD – Manage User

This start with the user chooses to create a new User or to edit an existing user. The user is sending a message to the ManageUserController (username, initials | | initials). The ManageUserController now sends the parameters further to the UserCatalog create or edit. The userCatalog now sends the parameters further to the UserDAO. Then the UserDAO ask the DBHandler for a connection to the Database. When the connection is established, the UserDAO sends the username, initials or just the initials to the Database witch is returning the matching username for the initials or creating a new user with the parameters. The username is now returned and presented for the user all the way back to the GUI.



SD – Login

This start with the user sending a message to ManageUserController (initials). The ManageUserController now ask the UserCatalog for the username matching the initials. The UserCatalog ask the UserDAO to get the Actual username. The UserDAO now gets a connection to the Database from the DBHandler. When the connection is established, the UserDAO sends the initials to the Database witch is returning the matching username for the initials. The username is now returned and presented for the user all the way back to the GUI.

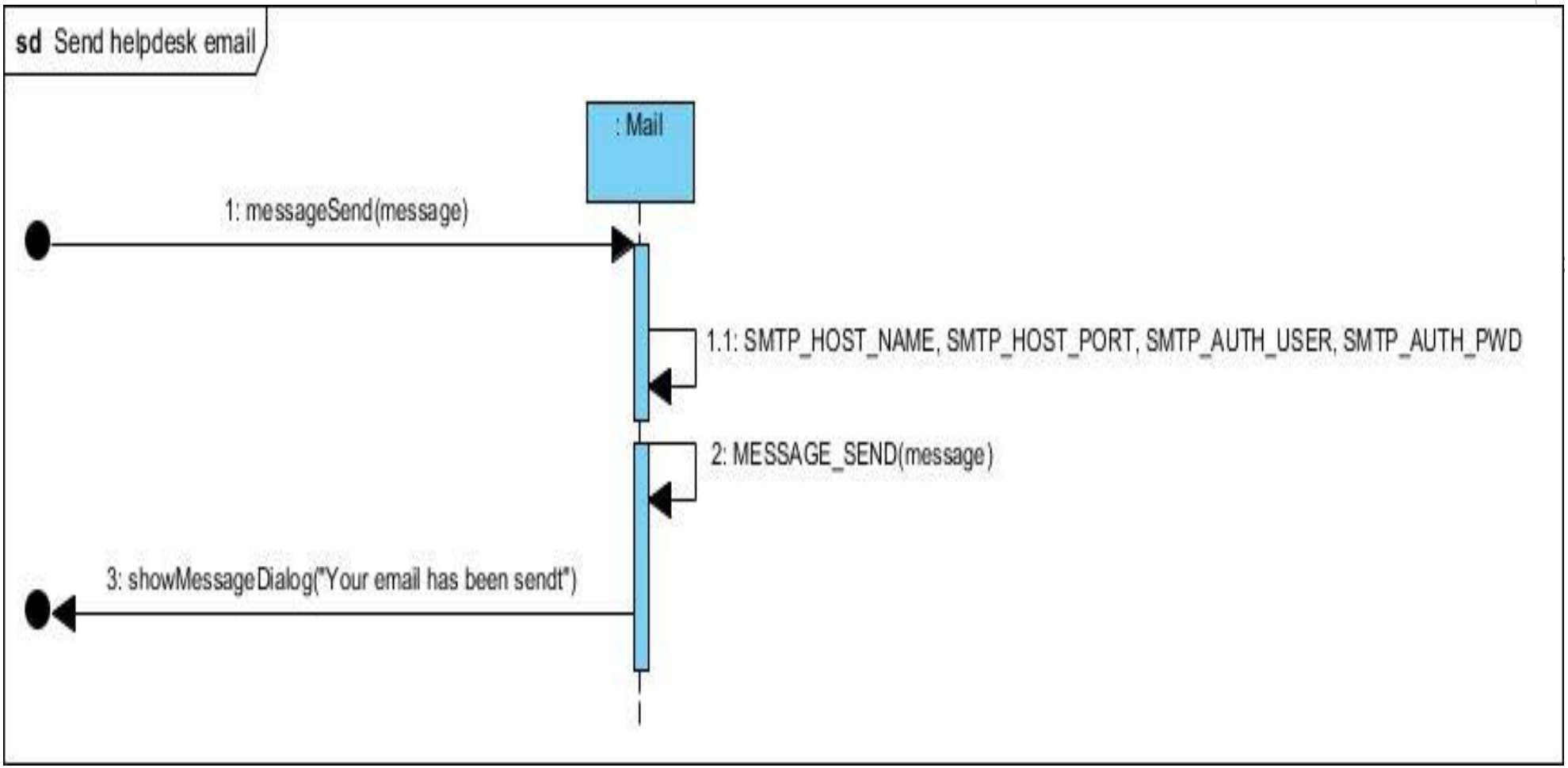


SD – Login

SD – Send Helpdesk Email

This start with the user sending a message to Mail with parameters (message) which is the text he want to send. The mail object now calls the external SMTP server and sends the email settings and the message.

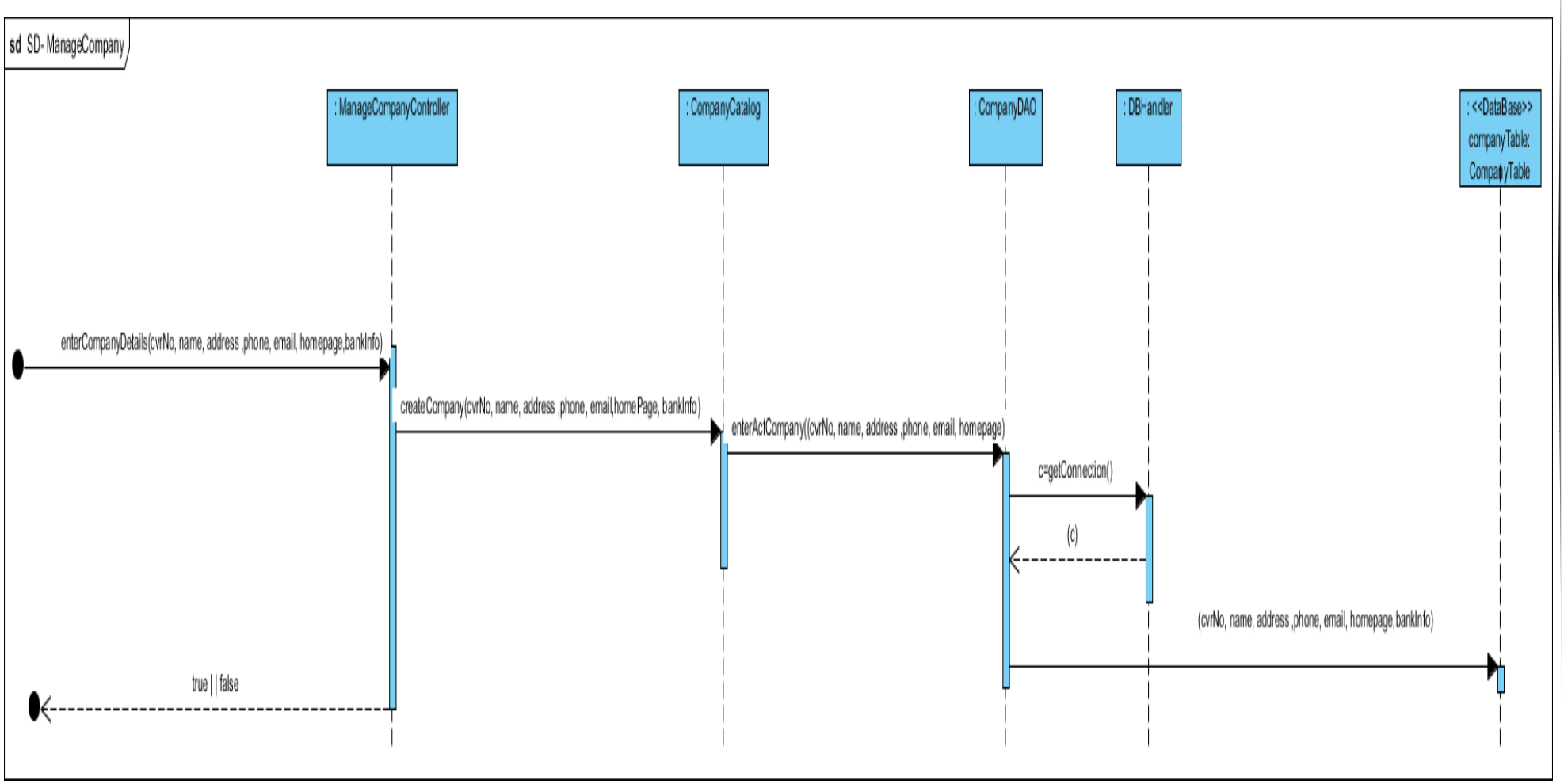
the external SMTP server is sending the email and responding with a true if send and false if not.



SD – Send Helpdesk Email

SD – Manage Comapny

This start with the user sending a message to ManageCompanyController with parameters (controller pattern). The controller now sends the parameters(company details) further to the CompanyCatalog. The CompanyCatalog now sends the parameters furher to the CompanyDAO – wich calls DBHandler for getting a conection to the database. When the connection is established the CompanyDAO sends the parameters to the Database

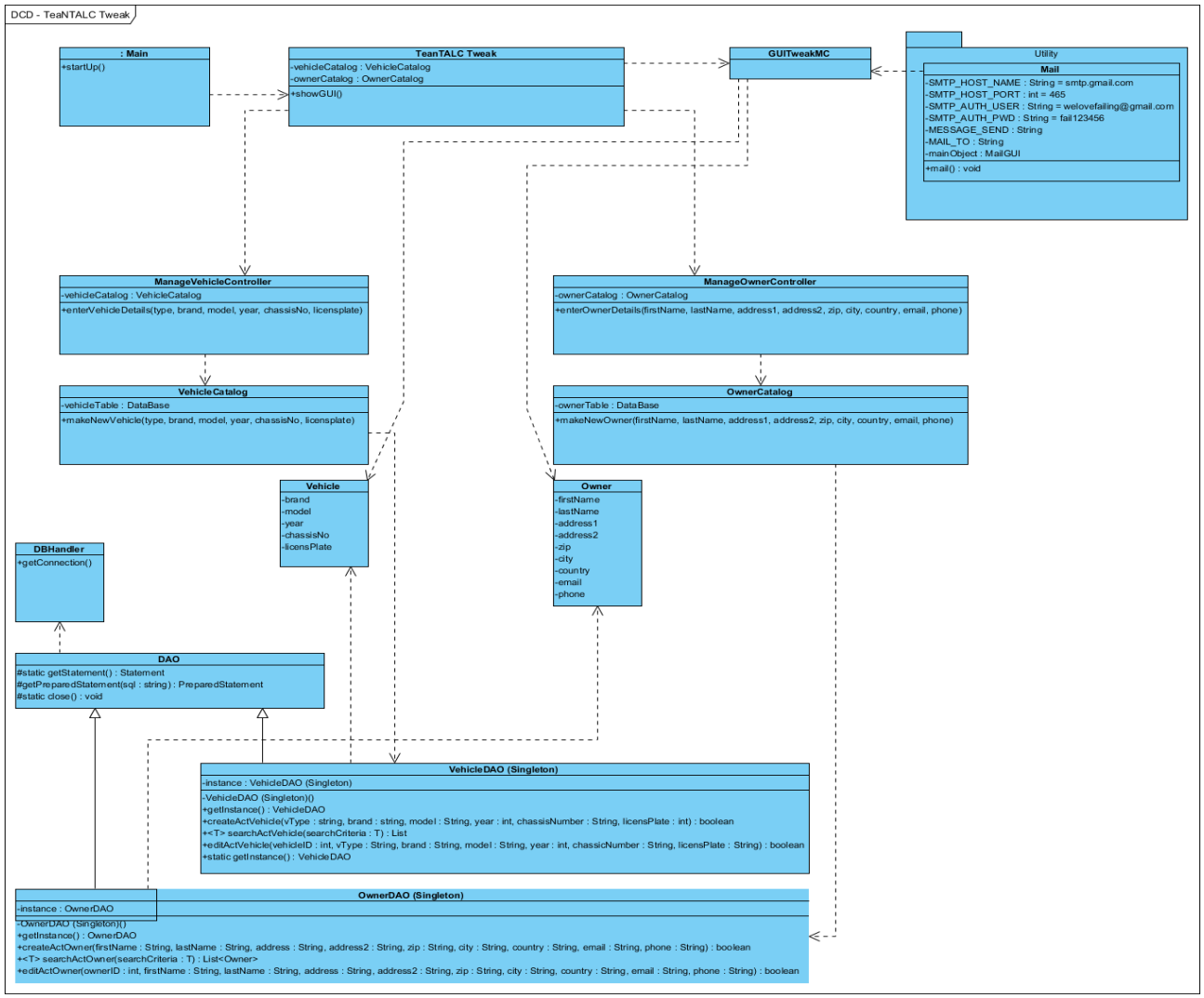


SD – Manage Company

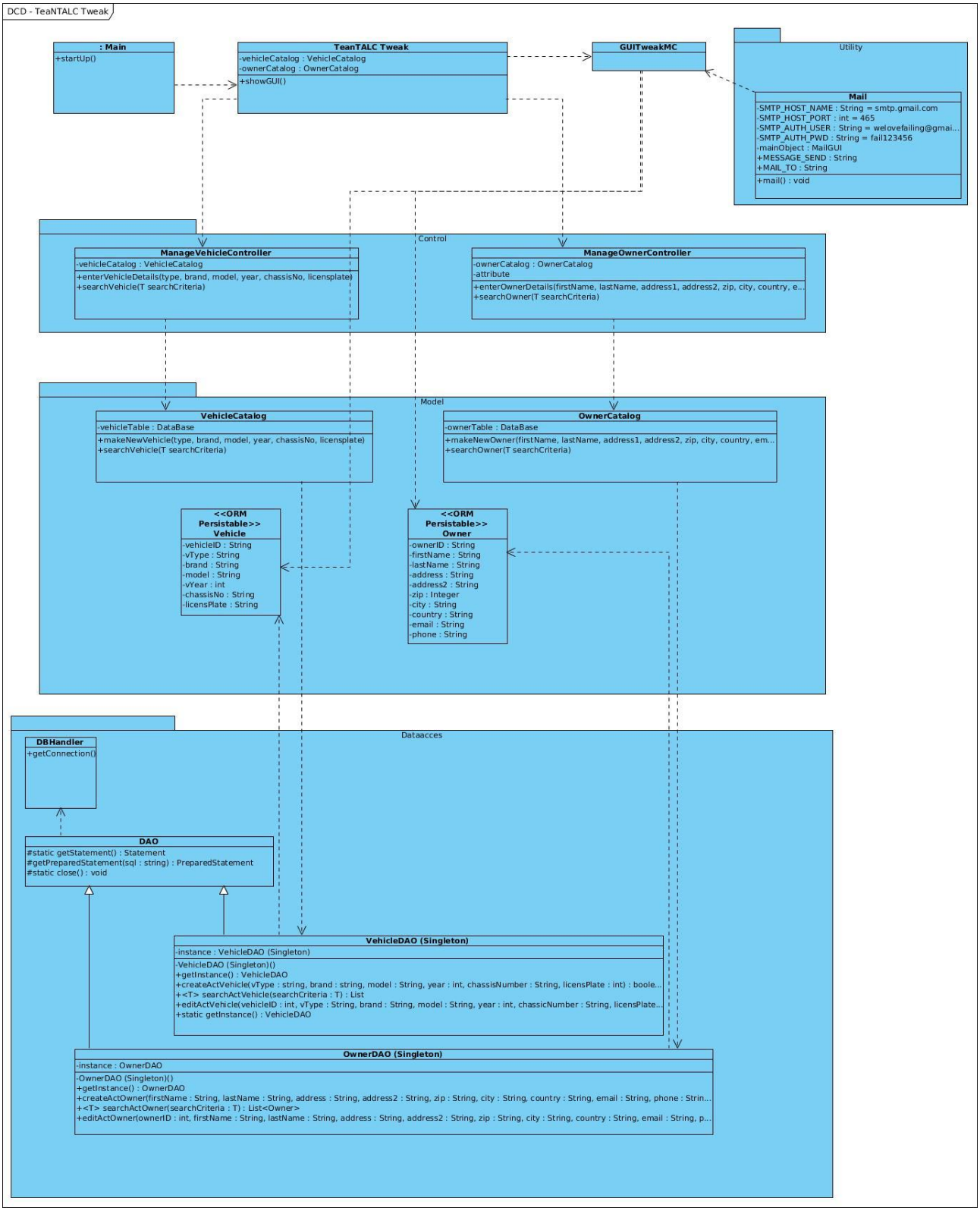
Appendix J – DCD

There will be attach a complete DCD in big format

Construction 1. Iteration



Construction 2.+3. Iteration



Appendix L – JAVA Code

Here we will show our most “exciting” code snippets

Glossary

Word:	Deskription:
Address	Street name and number of owner Legal: letters and numbers, empty Illegal: special signs
Address2	Street name and number of owner Legal: letters and numbers, empty Illegal: special signs
City	City for owner Legal: letters, empty Illegal: numbers, special signs
Country	Country there the owner lives Legal: letters, empty Illegal: numbers, special signs
Email	Email address for the owner Legal: letters, numbers, special Signs, empty, should follow the regular expression: start[A-Z0-9._%+-]+@[A-Z0-9.-]+\.[A-Z]{2,4}end Illegal: all other
Empty	Nothing fill in field
First Name	First and middle name of the owner of a vehicle Legal: letters and numbers Illegal: special signs, empty
Last Name	Last name of the owner of a vehicle Legal: letters and numbers Illegal: special signs, empty
Letters	a-å and A-Å
Moveable media	Storage place for a report, can be USB , DVD or CD
Numbers	0-9
Owner	Owns a vehicle known in the shop
Phone	Phone number for the owner Legal: numbers Illegal: empty, letters, special signs, more or less than 8 digits if country is Denmark
Report	The documentation for the work done on a vehicle, can contain comments, movies, pictures, sound, uploaded files
Shop	The place there optimizing and repairing is done
Special Signs	!”#¤%&/()=?*’¨´| -<>\£${[]}€@
Vehicle	At the moment there are 2 types of vehicle: MC(Motorcycle) and ATV(4 wheel MC)
WhiteBoards	“Boards” there user can type in measurement about the optimizing of the vehicle
Zip	Zip code for owner Legal: letters, numbers, empty, Illegal: special signs, letters

	if country is DK special criteria will be met: DKLegal.:4 digits, 9998 >= ZIP => 0100, numbers, empty DK Illegal: letters, special signs

Code standard

As we are 5 in the team and we all have to program different parts of the system, there´s of course 5 different ways of coding. In this chapter we will make some standards so our different minds will work a little more alike.

When we use

Names

- All methods and variables must be in English
- Method names must describe what the method does
- Variable names must describe what the method does
- camelCasing must be used for variable- and method names

Java doc

The documentation is very important for others viewing the code, even the other team members or even the coding member – in some cases. Therefore the java doc must be created while the specific method is done.

Template for classes

Author of each class must use this predefined author sign where ”name” is changed to the person in charge of the class.

```
/**
   * @author ”name” / TeaN TALC
   * @date Oktober – December 2010
   */
```

Metod-skeleton

The curly bracers bounding the methods must be below the method signature as shown below.

When a “while”, “if”, “else”, “for” etc. is ending, a comment containing “end n” should be placed (fx. //end if). This will make the code more orderly and easier to brows for the viewer, also shown below.

```
public VehicleremoveVehicle(int id)
{
    Vehicle vehObjekt = null;
int index = 0;
boolean found = false;
while(index <vehicleList.size() && !found)
    {
            vehObjekt = vehicleList.get(index);
            if(vehObjekt.getiD() == id)
            {
                    found = true;
                    vehicleList.remove(index);
            }//end if
            else
            {
```

```
                index++;
                }//end else
}//end while
return null;
}
```

Construction 1 iteration – Mail - Nyvang

```
packagetweakmc.utility;

/**
 *
 * @author Nyvang
 */
importjavax.mail.*;
importjavax.mail.internet.*;
importjava.util.Properties;
importtweakmc.view.MailGUI;


public class Mail {

private static final String SMTP_HOST_NAME = "smtp.gmail.com"; // SMTP hostname of the email account
to send from, here gmail (until a WLF mail has been created)
private static final int SMTP_HOST_PORT = 465; // gmail port
private static final String SMTP_AUTH_USER = "welovefailing@gmail.com"; // userlogin to gmail
private static final String SMTP_AUTH_PWD  = "fail123456"; // password
public static String MESSAGE_SEND; //variable for the send message text from the GUI ←special
public static String MAIL_TO; // variable for specifying an email address - just because i can :) ←special
privateMailGUImainObject; //mailGUI object for calling the showOKdialog method


public void mail() throws Exception{
      Properties props = new Properties();
props.put("mail.transport.protocol", "smtps");
props.put("mail.smtps.host", SMTP_HOST_NAME);
props.put("mail.smtps.auth", "true");
      // props.put("mail.smtps.quitwait", "false");

      Session mailSession = Session.getDefaultInstance(props);
mailSession.setDebug(true);
      Transport transport = mailSession.getTransport();

MimeMessage message = new MimeMessage(mailSession);
message.setSubject("Technicial error report for TweakMC"); // subject of the email
message.setContent(MESSAGE_SEND, "text/plain"); // message to send

message.addRecipient(Message.RecipientType.TO,
newInternetAddress(MAIL_TO)); // who the fuck should receive the mail?

transport.connect
       (SMTP_HOST_NAME, SMTP_HOST_PORT, SMTP_AUTH_USER, SMTP_AUTH_PWD); // it all in the
blender and send it to the cloud

transport.sendMessage(message,
message.getRecipients(Message.RecipientType.TO));
```

```
transport.close();
mainObject.showOKdialog();

   }
}
```

Appendix M - Test

Below are the different tests we have done.

Test case: Register Owner

System name: **Tweak**[MC]	Phase: **Inception**		
Test case: **Register Owner**			
Risk: **Medium** Extend: 1. Register a new Owner with valid and invalid values in fields. 2. If the user tries to push the "Save" button every field is checked and the status label will tell what's wrong.			
Test cases**First Name**	Equivalence	Expected	Actual Result
T1: "! Lars 32 " – User push "save" button	*First name may not contain signs*	First name field turns red	First name field turns red (see appendix Test)
T2 : " " – User push "save" button	*First name may not be empty*	First name field turns red	First name field turns red (see appendix Test)
T3 : " Lars 23 " – User push "save" button	*First name may contain letters and numbers*	OK	First name field stays white (see appendix Test)
Done by: Lars	**29/11-2010**	JUnit reference: **see Figure 3 14**	
Accepted by: Lars	**29/11-2010**		

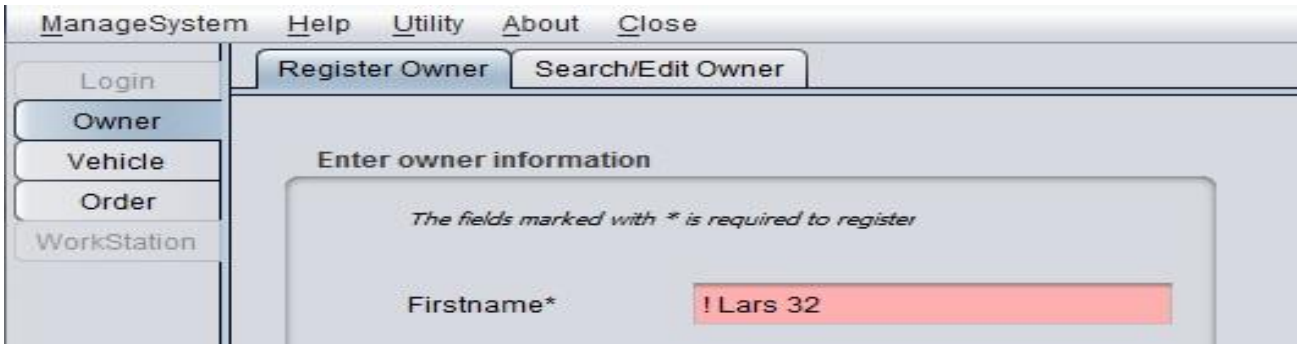


Figure 1 - Test case 1 ;First name is Invalid

Figure 2 - Test case 2 ;First name is Empty

Figure 3 - Test case 3 ; First name is Valid

System name: **Tweak**[MC]	Phase: **Inception**		
Test case: **Register Owner**			
Risk: **Medium** Extend: 1. Register a new Owner with valid and invalid values in fields. 2. We have a 'focus lost listener' that will highlight the field if something's wrong 3. If the user tries to push the "Save" button every field is checked and the status label will tell what's wrong.			
Test cases**Zip**	Equivalence	Expected	Actual Result
T4 : "98" – User push "save" button	*Zip should be between 100 and 9998*	Zip field turns red	Zip field turns red (see appendix Test)
T5 : "9999 " – User push "save" button	*Zip should be between 100 and 9998*	Zip field turns red	Zip field turns red (see appendix Test)
T6 : " 100 " – User push "save" button	*Zip should be between 100 and 9998*	OK	Zip field stays white and city field shows city (see appendix Test)
Done by: Lars	**29/11-2010**	JUnit reference: **see Figure 3 14**	
Accepted by: Lars	**29/11-2010**		

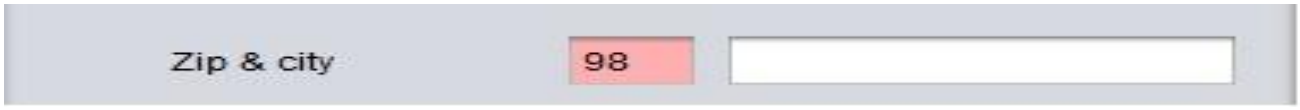
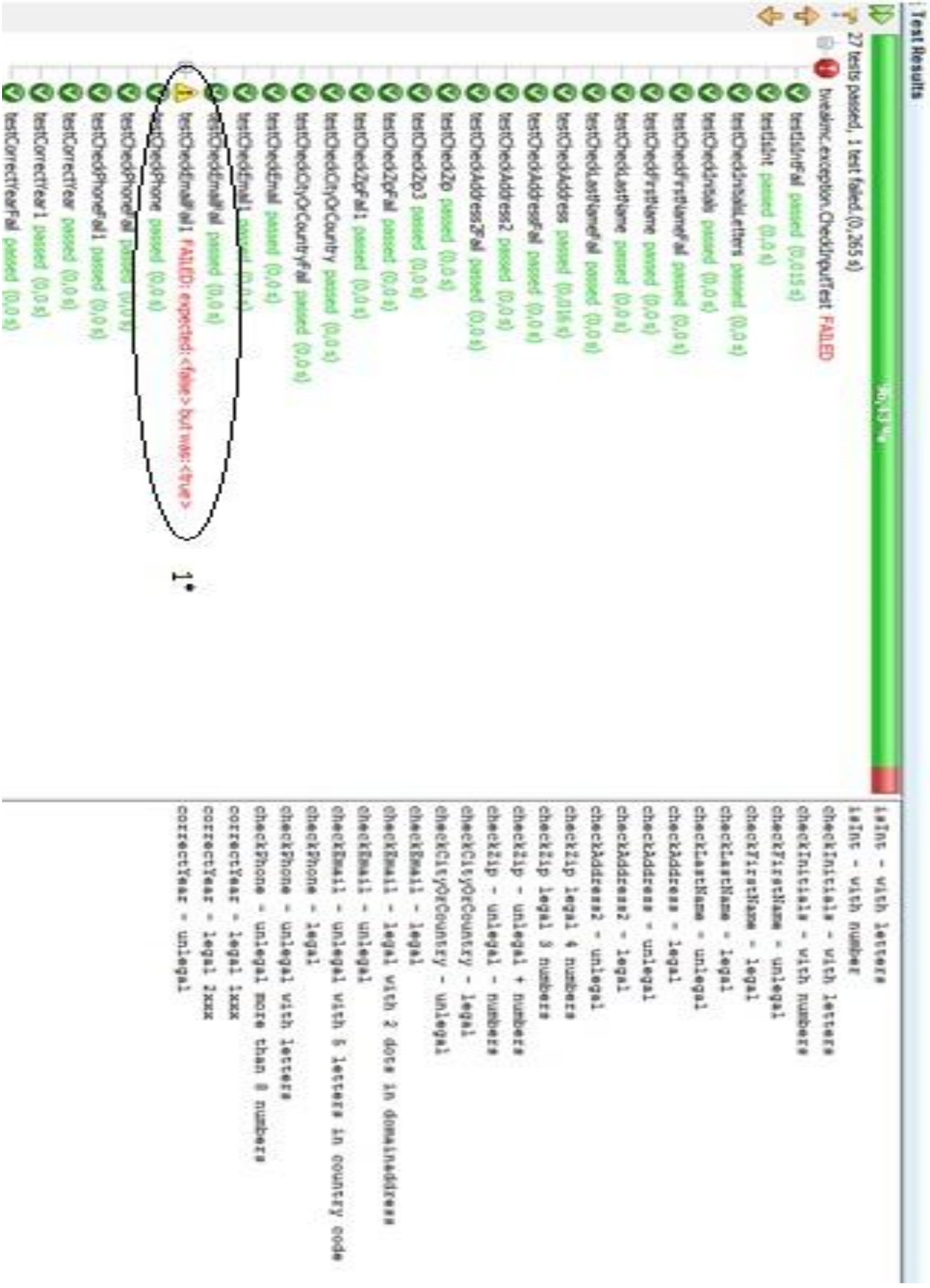


Figure 4 - Test case 4 ; Zip is invalid

Figure 5 - Test case 5 ; Zip is invalid

Figure 6 - Test case 6 ; Zip is Valid


Test Results
27 tests passed, 1 test failed (0,265 s)
testIsInt - with letters
testIsInt - with number
testCheckInitials - with letters
testCheckInitials - with numbers
testCheckFirstName - unlegal
testCheckFirstName - legal
testCheckLastName - legal
testCheckLastName - unlegal
testCheckAddress - legal
testCheckAddress - unlegal
testCheckAddress2 - legal
testCheckAddress2 - unlegal
testCheckZip legal 4 numbers
testCheckZip legal 3 numbers
testCheckZip - unlegal + numbers
testCheckZip - unlegal - numbers
testCheckCityOrCountry - legal
testCheckCityOrCountry - unlegal
testCheckEmail - legal
testCheckEmail - legal with 2 dots in domainaddress
testCheckEmail - unlegal
testCheckEmail - unlegal with 5 letters in country code
testCheckPhone - legal
testCheckPhone - unlegal with letters
testCheckPhone - unlegal more than 8 numbers
testCorrectYear - legal 1xxx
testCorrectYear - legal 2xxx
testCorrectYear - unlegal
1.

Appendix N – Considerations

To document the decision process we agreed to make notes of our discussions. This is mainly because the customer, KJ Motorcycles, have a better understanding for some of the decisions we have made throughout the project.
Even though we find this chapter very important as documentation, we cannot let the note taking affect the discussions we have as min. one team member must be dedicated to taking notes and therefore cannot participate in the discussion in the same way as the others. And as it is important for the team, that all team members are participating in these discussions, only a part of the discussions will be documented in this chapter.

Mockup

As the company specializes in a quite special area with some very special values, we had some issues quite early in the project. When we had to do the use cases we had to know which measurements are taken from which stations and what parts they are bound to. Therefore we decided to make a mockup and show it to the company. By doing this, we could get the different variables, data types etc. for the different stations. This make it easier to do the diagrams, and therefore easier to create the different classes. Besides, it´s always an advantage to have the end user(s) on the sideline when doing a project, despite this, many team´s often forgets to involve the users.

How should "makeReport ()" work

Since the system should present the result of all the work done on a vehicle, we have to decide in which way to present it for the user, so this is just thoughts of how to get it working and not yet a use case.

The report has to be created, of course. The question is how the report should be created because there´s some issues with images sound, and video. The standard output for Java (from what we have learned) is a *.txt file and this cannot contain media. We decided that there should be several options when generating the final report. The user should be able to get a printable copy and an electronic copy (with media).

We´ve come up with some different solutions for this;

PDF files:

1. Pro´s
    a. Easy to handle
    b. Readable on any PC or Mac
    c. Size is somewhat small
2. Con´s
    a. Difficult to make with our limited knowledge about Java
    b. Can be very time consuming

TXT files

1. Pro´s
    a. Very easy to generate
    b. Very small size
2. Con´s
    a. No graphical options
    b. Difficult to manage layout

HTML

1. Pro´s
    a. Easy managing layout
    b. Can contain all media

    c. Easy to distribute
    d. Java have a tight relationship to html
2. Con´s
    a. Lack of knowhow
    b. c*an* be difficult to print

We all agree that there is a high risk of this taking too much time, and the main goal is to produce a report with measurements, that is in text format. However, as we intend to learn as much as possible we will investigate whether either *.pdf or another format is possible within the timeline and if we can find a premade class that have this property.
*****Ongoing discussion*****

Time

We talked about time of the different bikes. Can we make a time frame that counts the hours spend on the individual order.

Should it be a fixed time table or should it start when the project is created and stops at the end of each day? Or should the mechanic manually start and stop the time when he´s working on the project?

We have to consider that the program needs to lighten their daily work and not to give them extra workload.

Therefore we have decided to create 2 functions, one is used for editing/working on a project (working on whiteboards), and the other is used for viewing the project ("make report()"). In the editing option the time will start automatic and the view option is just for viewing. The risk of this is if the user forgets to close the system after each day or when the project is finished.

This must be the best solution due to the workload issue, and it should be accurate enough to measure the actual time spend on each project. But since the project price is fixed and therefore not dependent on the actual time spend this function is just for internal use and therefore it is up to the company if they want to use the function.

The login class

To maintain the low cohesion and high coupling, we decided to make a separate login class, who should be responsible for handling the user authentication.

This however gave us quite some issues that we were a long time to solve.

As we are using tabbed panes and we want them to be disabled until successful login, we bashed our heads against a great wall. For some reason, the "setEnabledAt" method cannot be called from another class, like the Login class, and used for enabling the panes containing e.g. vehicle or owner. It was only possible to set the panes enabled or disabled before the JFrame was set to visible, we cannot change it when the system is running.

We searched a long time for a solution, but we had little luck. We came up with 2 alternative solutions:

1. Tabbed panes are history, and we would use ordinary panes and just set the buttons who calls the panes to be enabled or disabled. Which we have tested in an earlier application, and we know how/that this works.
2. A new JFrame should be created, that opens the mainUI after successful login. By doing this, the tabbed panes could be enabled from the beginning in the mainUI.

Discussing this, we accidentally tried calling the “setEnabledAt” method from inside the mainUI class. This got the job done. We could now enable and disable the tabbed panes as we wanted to.

So the decision is that we would drop the Login class and break the low cohesion and high coupling design patterns.

Appendix O - Risklist

We have placed the remaining risks in this section of the appendix.

<RSK-04> Real-time Constraints (Time)			
Magnitude	**Description**	**Impact**	**Strategy plan**
5	If we meet constraints suddenly because of our experience.	H	Make sure time is available in our plan to deal with constraints

<RSK-05> Commitment (People)			
Magnitude	**Description**	**Impact**	**Strategy plan**
3	Even though we are in class don’t mean we “commit”	H	Read the agreed stuff, so everybody is committed in the actual discussions

<RSK-06> Backup and Restore (Technical)			
Magnitude	**Description**	**Impact**	**Strategy plan**
1	If something goes wrong it’s important we have backup of our project	C	Backup system is running and backing everything up twice a day

<RSK-07> Team Spirit (People)			
Magnitude	**Description**	**Impact**	**Strategy plan**
1	Team spirit is important to keep high so all in the team is feeling secure.	C	Do socialize things beside the project like go out and eat together or other things to socialize

<RSK-08> Lack of Experience (Production)			
Magnitude	**Description**	**Impact**	**Strategy plan**
5	Lack of programming experience is for all of us – we have not tried it that much	M	Try different solutions – and see other team member’s solutions.

<RSK-09> Experience (Planning)			
Magnitude	**Description**	**Impact**	**Strategy plan**
4	Our experience in planning is very low because we haven’t done it before.	M	Measure how long time everything is running – so we keep being better to plan more precisely

<RSK-10> Date of Delivery (Schedule Risks)			
Magnitude	**Description**	**Impact**	**Strategy plan**
1	We try to be finished with the product to the date of delivery	C	Be ready for 'overtime'.

<RSK-11> Availability (People)			
Magnitude	**Description**	**Impact**	**Strategy plan**
2	Team members unavailable for overtime	H	Lower our goals

<RSK-12> Schedule (Time)			
Magnitude	**Description**	**Impact**	**Strategy plan**
2	Our planning can be badly scheduled.	H	Review the schedule in every iteration

<RSK-13>Production (Funding)			
Magnitude	**Description**	**Impact**	**Strategy plan**
1	External description[1]	L	Alternative we will use an external company for printing or the schools pressroom. And make the DVD´ s ourselves

[1]*Since this is a study project, funding is not the main issue but however as we aim to do the transition phase, then this may cost us some money for producing hardcopy of the project for example DVD's.*
Since there is so many people using the schools printers and there is a lot to print and we are limited by time – we may use external means in order to cope up.

Appendix P– project plan

	ID	Name	Start	Finish	Name
	P	Project	17-09-2010	16-12-2010	
	pe	Project Establishment	17-09-2010	26-09-2010	Nicolaj
					Lars
					Adams
					Torben
					Claus
	Incep	**Inception**	**27-09-2010**	**11-10-2010**	
	incepA	Inception	27-09-2010	08-10-2010	Nicolaj
					Lars
					Adams
					Torben
					Claus
	incepM	Lifecycle Objectives Milestone	11-10-2010	11-10-2010	
	elab	**Elaboration**	**11-10-2010**	**15-10-2010**	
	elab1	Elaboration 1. iteration	11-10-2010	13-10-2010	
	elab2	Elaboration 2. iteration	14-10-2010	15-10-2010	
	elabM	Lifecycle Architecture Milestone	15-10-2010	15-10-2010	
	conc	**Construction**	**25-10-2010**	**03-12-2010**	
	conc1	Construction 1. iteration	25-10-2010	29-10-2010	
	conc2	Construction 2. iteration	01-11-2010	05-11-2010	
	conc3	Construction 3. iteration	08-11-2010	12-11-2010	
	conc4	Construction 4. iteration	15-11-2010	19-11-2010	
	conc5	Construction 5. iteration	22-11-2010	26-11-2010	
	conc6	Construction 6. iteration	29-11-2010	03-12-2010	
	concM	Initial operationel capability Milestone	03-12-2010	03-12-2010	
	tran	**Transition**	**06-12-2010**	**14-12-2010**	
	tran1	Transiotion 1. iteration	06-12-2010	10-12-2010	
	tran2	Transiotion 2. iteration	13-12-2010	14-12-2010	
	tranM	Product Release Milestone	14-12-2010	14-12-2010	
	1	Nicolaj			
	2	Lars			
	3	Adams			
	4	Torben			
	5	Claus			

The second version of the project plan. As we fell behind, we had to revise the plan.

Appendix Q – Construction 5th iteration - Adams

Iteration 5 Use case 15 – Manage Company

UC15: Manage Company
Scope: Tweak[MC]
Level: User goal
Primary actor: User
Stakeholders and interests:
User wants to save or edit company information
Preconditions: Information to be saved is available.
Post conditions: Company information is saved

Main Success scenario:

1. Need for user to save or edit company's information
2. User starts manage company
3. System presents company information if available
   (If information available go to no.4 if not go to no.)
4. User edit available information
   (User goes to no.6)
5. User enters basic company information (cvrNo, name, address ,phone, email, homepage, bankInfo)
6. System saves Company's information
7. User end manage Company

Alternative flows:

a) At any time system fails:
   1. User restarts system, and request recovery of prior state.
   2. System reconstructs prior state
   3. Users start a new recording and retrieve the information.

b) User types in wrong information
   1. User resets fields and try again

c) System fails to save
   1. User restarts system
   2. System restarts
      2a System fails to save again.
         1. User restarts computer and goes to step 1
   3. System saves company information

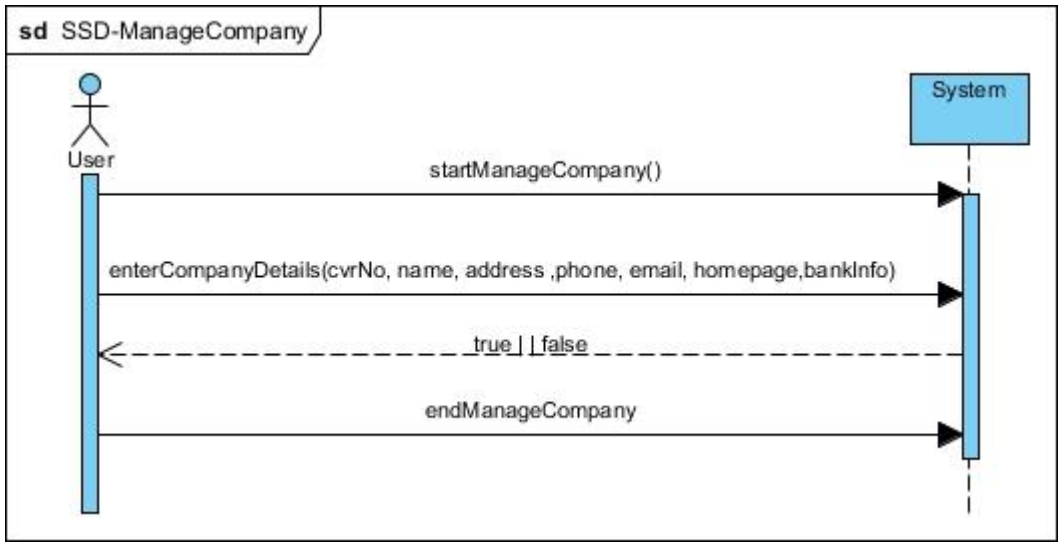
d) At any time system send error message:
   1. User press F1 and are shown Help Menu
   2. User read and tries understanding information in Help Menu
      2a. User do not understand Help Menu
         1. User sends email via help desk function
      2b. User understands and goes to step 3
   3. User tries again

Special requirements: none
Technology and Data Variations List: none
Frequency of occurrence: approximately twice a year
Open issues: none

Use Case Mange Company is as important as other crucial Use Cases in this project, simple because without the Company then other Use Cases may be would not exist. User would like to save or edit the information of the Company; User starts the ManageCompany and enters the basic information of the company if the information is correct then the system will save the information. On the other hand if user enters wrong information, information will not be saved and this means that the user will start afresh until he or she enters the correct information, and he will end the process at the end. Ref appendix.

System Sequence Diagram

System Sequence Diagram above explains the ManageCompany. User starts ManageCompany and he enters the Company's details (cvrNo, name, address, phone, email, homepage, bankInfo) then the system will check if it is the right information, information will then be saved if it is correct and if the information entered is not correct then the system will return false sign, then the user has to restart again until he provides right information. Ref diagram appendix.



Operation contract

Operation contract of ManageCompany it has the four properties an Operation Contract should have and this includes the name, cross reference, pre condition, and post condition. The connection to the data base is formed in the pre condition and in the post condition company was edited However in the operation the company information is also needed in order to access the company and this is shown in the operation contract. Ref appendix for the diagram.

Operation Contracts ManageCompany

TeaN TALC – Motorbike application

OC15: enterCompanyInfo (cvrNo, name, address, phone, email, homepage,bankInfo)

CrossRef: Manage Company

Pre Conditions:

A connection to database is established

Post conditions:

-company was updated

Architecture analysis of UC-15 ManageCompany

The architecture shows the connection of different classes in different packages right from the Mail GUI to the DataBase. MainGUI is connected with the CompanyGUI and this connects to the ManageCompanyController this calls the CompanyCatalog and the catalog creates the company and the CampanyCatalog calls the CompanyDAO and this connects to the DAO to get data about the Company and this connects to the Database.
Through such a process we are able to update and edit the company and save the information. Reference Appendix for architectural design

This Use case is relevant in a way that without any company then other classes related to a company cannot exist. To me I think this is the back born of our project
May be we could not have the company catalog since the company already exists so we could only ask for their Database and then we just add what we want to add.
I think it's better to take it separately without combining it with another use case. Ref appendix for the UseCases.

GUI Design has some important futures of which include edit, save and the parameters like name , address, phone, email ,homepage ,cvrNo and bankInformation, all of which are vital.

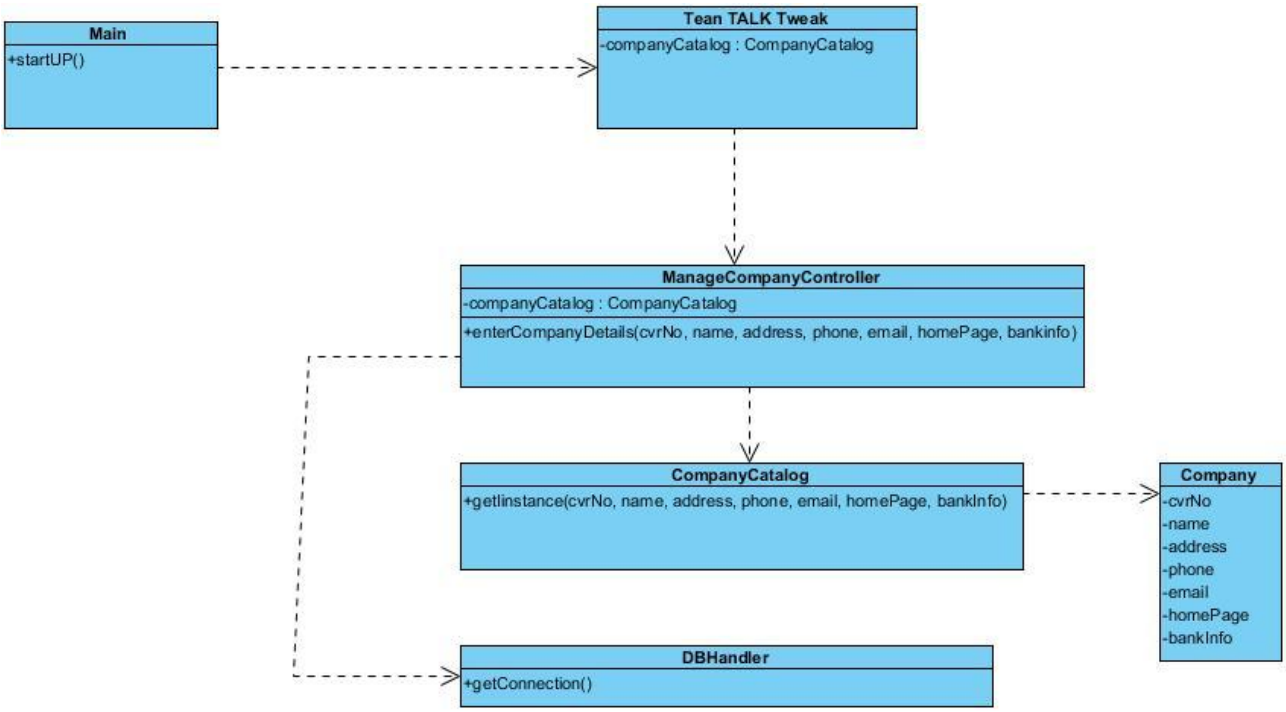
Sequence Diagram

sequence Diagram has Company GUI where all the parameters of the company are entered by the use of Company controller, and this will call the Company catalog and this is responsible of the creation of the

company, this will call the company DAO by the use of DataHandler and this has a method getConnection (); this will connect to the Database and then Data is accessed. Ref appendix for the sequence Diagram.

The DesignClassDiagram

The diagram shows the connection programming. The mail is started and the ManageCompanyController will be called to call for CompanyCatalog which has the Company instance and ManageCompanyController call s DBHandler which has the getConnection, to connect to the DAO.



CLASS COMPANY CODE

```
public class Company {

   private String cvrNo;
   private String name;
   private String address;
   private String phone;
   private String email;
   private String homePage;
   private String bankInfo;
/**
 * constructor for the company
 * @param cvrNo for the company
 * @param name of the company
 * @param address of the company
 * @param email for the company
 * @param homepage for the company
 * @param bankInfo for the company
 */
```

```
   public Company(String curNo, String name, String address,String phone,String email, String homePage,
String bankInfo)

      {
       this.cvrNo = curNo;
       this.name  = name;
       this.address = address;
       this.phone = phone;
       this.email = email;
       this.homePage = homePage;
       this.bankInfo = bankInfo;


      }
   /**
    * methord returns company cvrNo
    * @return
    */

   public String getCvrNo()
      {
          return cvrNo;
      }
   /**
    * Methord sets cvrno for the company
    * @param cvrNo
    */
   public void SetCvrNo(String cvrNo)
   {

   }
   /**
    * Methord sets the company name
    * @param name
    */

   public void setName(String name)

      {
          this.name = name;
      }
    /**
     * Methord returns company name
     * @return
     */
   public String getName()
```

```
    {
        return name;
    }
/**
 * Methord to set company address
 * @param address
 */
public void setAddress(String address)

   {
       this.address = address;
   }
/**
 * Methord retuns company address
 * @return
 */
public String getAddress()

  {
     return address;
  }

/**
 * Methord sets company phone
 * @param phone
 */

public void setPhone(String phone)
   {
     this.phone = phone;
   }
/**
 * Methord returns phon number of the company
 * @return
 */
public String getPhone()
   {
     return phone;
   }
/**
 * methord returns email
 * @param email
 */
public void setEmail(String email)
   {
     this.email = email;
   }
```

```
/**
 * Methord returns email
 * @return
 */
public String getEmail()

   {
      return email;
   }
/**
 * Methord sets homePage
 * @param homePage
 */
public void setHomePage(String homePage)

  {
      this.homePage = homePage;
  }
/**
 *
 * @return
 */

public String getHomePage()

  {
      return homePage;
  }
/**
 * Methord set bankIfo of the company
 * @param bankInfo
 */

public void setBankInfo(String bankInfo)
  {
      this.bankInfo = bankInfo;
  }
/**
 * return bankInfo of the company
 * @return
 */
public String getBankInfo()
  {
      return bankInfo;
  }
/**
 * Return all information
 * @return
```

```
   */
   @Override
  public String toString()
   {
        return cvrNo + " " + name + " " + address + " " + phone + " " + email + " " + homePage + " " +  bankInfo
+ " ";
    }
   //End of company Class
}
```

This code shows the different variables of a company class and how it is created. All its fields are of type String It has also a constructor and some methods that helps in the setting and getting the Variables. This is the foundation of all other classes in different layers about the company for instance CompanyDAO, CompanyGui and others

Appendix R

TweakMc Order preview

Report generated by: nn119171, Tue Dec 14 18:28:52 CET 2010

Prewieving order: C27

Order type: 0
Start date: 20101207
End date: 20110226
Description:
Spareparts required:
Price estimate:
Vehicle ID: V20