Internship Australia
Developing at Nissan Motor Company
## Document information

### Version history

<table>
<thead>
<tr>
<th>Version</th>
<th>Status</th>
<th>Date</th>
<th>Description</th>
<th>Approved</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>Draft</td>
<td>September 30, 2010</td>
<td>Start document</td>
<td></td>
</tr>
<tr>
<td>0.2</td>
<td>Draft</td>
<td>October 3, 2010</td>
<td>Added Problem description, company background, main goal, learning skills</td>
<td>Oct 4, 2010</td>
</tr>
<tr>
<td>0.3</td>
<td>Draft</td>
<td>October 6, 2010</td>
<td>Added acknowledgement, start of system analyses</td>
<td></td>
</tr>
<tr>
<td>0.4</td>
<td>Draft</td>
<td>October 7, 2010</td>
<td>Added system layouts, updated planning</td>
<td></td>
</tr>
<tr>
<td>0.5</td>
<td>Concept</td>
<td>October 10, 2010</td>
<td>Added summary, Application letter, Application for graduation assignment and updated system analyses</td>
<td></td>
</tr>
<tr>
<td>0.6</td>
<td>Concept</td>
<td>October 11, 2010</td>
<td>System details updated</td>
<td>Oct 11, 2010</td>
</tr>
<tr>
<td>0.7</td>
<td>Concept</td>
<td>October 13, 2010</td>
<td>Reorder of chapters</td>
<td></td>
</tr>
<tr>
<td>0.8</td>
<td>Concept</td>
<td>October 17, 2010</td>
<td>Added Quality control and migration of the system</td>
<td>Oct 18, 2010</td>
</tr>
<tr>
<td>0.9</td>
<td>Concept</td>
<td>October 20, 2010</td>
<td>Update migration of the system</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>Concept</td>
<td>October 29, 2010</td>
<td>Update System analyses and literature list and added screenshots of the system</td>
<td>Oct 29, 2010</td>
</tr>
<tr>
<td>1.1</td>
<td>Concept</td>
<td>November 7, 2010</td>
<td>Sentence check and Migration of the system layout setup</td>
<td></td>
</tr>
<tr>
<td>1.2</td>
<td>Concept</td>
<td>November 15, 2010</td>
<td>Updated literature list with references in document</td>
<td></td>
</tr>
<tr>
<td>1.3</td>
<td>Concept</td>
<td>December 1, 2010</td>
<td>Updated Quality Control</td>
<td></td>
</tr>
<tr>
<td>1.4</td>
<td>Concept</td>
<td>December 13, 2010</td>
<td>Updated migration screenshots</td>
<td>Dec 14, 2010</td>
</tr>
<tr>
<td>1.5</td>
<td>Concept</td>
<td>December 26, 2010</td>
<td>Start of scenarios</td>
<td></td>
</tr>
<tr>
<td>1.6</td>
<td>Concept</td>
<td>January 5, 2011</td>
<td>Updated scenario images</td>
<td></td>
</tr>
<tr>
<td>1.7</td>
<td>Concept</td>
<td>January 23, 2011</td>
<td>Added process flow images to Migration</td>
<td>Jan 25, 2010</td>
</tr>
<tr>
<td>1.8</td>
<td>Concept</td>
<td>January 27, 2011</td>
<td>Added first part of metrics</td>
<td></td>
</tr>
<tr>
<td>1.9</td>
<td>Concept</td>
<td>February 28, 2011</td>
<td>Metric paragraph started and added CC complexity graph</td>
<td></td>
</tr>
<tr>
<td>2.0</td>
<td>Final</td>
<td>March 22, 2011</td>
<td>Added conclusion and ‘other tasks’ attachment</td>
<td></td>
</tr>
</tbody>
</table>
List of figures and tables

Figures
Figure 1: Nissan headquarters Australia .................................................................7
Figure 2: All Nissan establishments [W22] ..............................................................8
Figure 3: A general approach how the system is build up .......................................10
Figure 4: Applied MVC pattern on Java servlets ..................................................11
Figure 5: Basic flow diagram of the class layout ....................................................12
Figure 6: Nissan development process ..................................................................13
Figure 7: Menu of Company Cars .........................................................................15
Figure 8: Old declaration screen versus new declaration screen ............................15
Figure 9: FBT process ...........................................................................................16
Figure 10: FBT request email ................................................................................16
Figure 11: Completed FBT screen ...........................................................................16
Figure 12: Old add relation screen versus new add relation screen ......................17
Figure 13: Add relation process ............................................................................18
Figure 14: General part of Maintain relations ......................................................18
Figure 15: Filled Relative information screen .......................................................18
Figure 16: Old order screen versus new order screen .........................................19
Figure 17: Place order process .............................................................................20
Figure 18: Current vehicle details screen .............................................................20
Figure 19: Filled request order form .....................................................................20
Figure 20: Old simulate screen versus new simulate screen ...............................21
Figure 21: On / Off simulation screen ..................................................................21
Figure 22: Header with debug and simulation on ...............................................21
Figure 23: Wysiwyg editor for editing the ads .....................................................22
Figure 24: Old and new login screen .....................................................................22
Figure 25: Specials ad ..........................................................................................22
Figure 26: QA process enlarged ...........................................................................23
Figure 27: Cobertura Coverage example ................................................................26
Figure 28: Feedback item example .......................................................................27
Figure 29: Kiviat Metrics Graph of Company Car System .................................28
Figure 30: Complexity graph ...............................................................................29
Figure 31: (Comment) Lines of Code graph .......................................................29
Figure 32: Fire rating sign, an example how the local government warns the people .30
Figure 33: Reload services screen .......................................................................42

Tables
Table 1: General SOAP request .............................................................................12
Table 2: General SOAP response ..........................................................................12
Table 3: General SOAP fault ...............................................................................13
Table 4: Message types .........................................................................................14
Table 5: Example function ...................................................................................24
Table 6: Tests written for example function .........................................................24
Table 7: Private variables UserDetails entity .......................................................25
Table 8: Getter / Setter test source ......................................................................25
Table 9: Implementation of excluded variables ....................................................25
Executive summary

This report contains the work done within the internship assignment of Arnoud Sanderse. The assignment includes the rebuild to a new environment of Nissan Company Car System Australia. Next to this main project, there will be several side projects to fill up the waiting time caused by the response time of outstanding requests.

Nissan Australia is a part of Nissan Global, a worldwide car company. The rebuild of the Nissan Company Car System will be done in the development department of Australia, which is responsible for building, maintaining and supporting all Nissan applications within Nissan Australia.

This document is written for the examiners of the Hogeschool van Amsterdam and developing staff of Nissan Australia. The author expects a minimum knowledge about computer technology and Java programming for understanding the document.

At the end of the document are a literature list and a glossary added for clarification of technical terms in the report. In addition, some attachments are added for more (background) information of the internship.

I would like to point out that my experience with Nissan Australia and there staff was very special for me and will thank them for their support, help and giving me the chance to do my internship with them.
# Table of contents

Document information ........................................................................................................................................... 2
  Version history..................................................................................................................................................... 2
  List of figures and tables ...................................................................................................................................... 3
Executive summary ............................................................................................................................................... 4
Table of contents .................................................................................................................................................. 5
Acknowledgements ............................................................................................................................................. 6
Introduction ......................................................................................................................................................... 7
  Problem description ........................................................................................................................................... 7
  The company ..................................................................................................................................................... 7
Skills to be learned and demonstrated ........................................................................................................... 8
Technical tools / plugins ...................................................................................................................................... 9
Planning ............................................................................................................................................................... 9
System analysis .................................................................................................................................................... 10
  General ............................................................................................................................................................ 10
  System overview ........................................................................................................................................... 10
Overview Company Car System ....................................................................................................................... 14
Migration of the Company Car System ........................................................................................................... 15
  General ........................................................................................................................................................... 15
  FBT Declaration ........................................................................................................................................... 15
  Add relation .................................................................................................................................................... 17
  Place order ..................................................................................................................................................... 19
  Admin ............................................................................................................................................................ 21
Secondary login .................................................................................................................................................. 22
Quality control ................................................................................................................................................... 23
  General ............................................................................................................................................................ 23
  Acceptance testing ........................................................................................................................................ 26
Project Metrics .................................................................................................................................................. 28
Conclusion ......................................................................................................................................................... 30
Literature list ....................................................................................................................................................... 31
  Web ................................................................................................................................................................. 31
  Books ............................................................................................................................................................ 33
Glossary ............................................................................................................................................................... 34
Attachments......................................................................................................................................................... 35
  I. Application letter internship .......................................................................................................................... 36
  II. Application for graduation assignment (Dutch) .......................................................................................... 37
  III. Company Car System complexity graph ................................................................................................. 38
  IV. General planning ...................................................................................................................................... 39
  V. Other tasks ............................................................................................................................................... 42
  VI. Program screenshots ................................................................................................................................. 43
  VII. Order Confirmation form ........................................................................................................................ 51
Acknowledgements

First and foremost, I would like to thank my dad for sharing his contacts in Australia and support during the application phase and also my direct family for their support and help.

Next, I would thank Nissan Australia for giving me a chance to do my internship at their company. Especially Steve Hogan (National Manager - Information Services & Solutions) and Jay Brunett (Account Manager - E-Business & Commercial Systems and my supervisor) for their help and believe that they could manage my internship. Also the development team I worked with, would I thank for their help and support. They showed me some examples that helped me setup the projects.

Besides, I would thank all teachers of the Hogeschool van Amsterdam for their support and help during my school career and internship. I’ve learned a lot, sometimes on the hard way, from them. I’m glad that I can conclude it was an unique and worth full experience.

Finally, I would thank the family Wilks for staying at their place the first months and their support. Also my family, friends and everyone I forgot for supporting and helping me during the setup and stay. Without help of all the people above, I would have faced many difficulties during this internship. Thanks their help I could do what I wanted to do for a long time.
Introduction

This internship report contains the results of the work during the internship at Nissan Australia\(^{[W8]}\). The internship assignment contains one large project called Company Car Order System. Next to this project, there will be several side projects to fill up the time gaps that are caused by response time of outstanding requests. These include projects like research of Java logging methods and improving general structure of the project setup. The assignment will be developed according to the developing standards of Nissan Australia. All projects within Nissan Australia must comply with these standards.

The internship is for the Software Engineering course of the Informatics department of the ‘Hogeschool van Amsterdam’\(^{[W9]}\) (University of applied sciences Amsterdam).

Problem description

Redevelop and enhance the current version of the Nissan Company Car Order System (further called Company Car System) into an Ajax enabled JSE\(^{[W20][W45][B1][B2]}\) Web application. The system is divided in 4 parts, namely: add relation, place order, declaration of vehicle data, simulate user and admin.

For each part there will be a analyse stage, building stage and test stage with a coverage test. After all the stages the tests will run again (with the coverage option) and the content manager will walk through the application and give feedback. After all these stages, the end user will test the application and give feedback over it.

The main goal for redeveloping and enhancing the old Company Car System is that the old system has lack of user friendliness and there are minor bugs in it what sometimes leads to problems. Also the implementation of Renault within the system and an uniform look of all applications within Nissan Australia was a reason to develop a new version.

The company

![Nissan headquarters Australia](image)

Figure 1: Nissan headquarters Australia

Nissan Motor Company \(^{[W8][W9][W50]}\) (further called Nissan) is a worldwide car company with its headquarters in Japan. The Australian part of Nissan is responsible for the whole Oceania. Oceania includes Australia and New Zealand.

\textit{Nissan founded in Yokohama City, Kanagawa Prefecture in December 1933, has sixteen (16) production sites in Japan and abroad and offers products and services in more than 160 countries worldwide. (Nissan Global)}
Since March 1999 Nissan and Renault are in the Renault-Nissan Alliance. With this Alliance they share technologies and innovative ideas to make their products the best quality without losing their separate brands and corporate identities.

The Australian IT-development-department is settled in Dandenong-South in the head office. The development team is responsible for maintaining, supporting and developing applications that are used by customers, dealers and employees.

![Figure 2: All Nissan establishments](image)

**Skills to be learned and demonstrated**

**Technical**
- Java / JSF
- Use and integration of Web Services
- XML technologies
- Reading and Interpreting other team members code
- Debugging problems

**Non-technical**
- Working in a collaborative Team environment
- Documentation
- Use and abiding to Standards
- Presenting work to team members in weekly project meetings

**Main learning goal**
Understand the business requirements for the Company Car System, and deliver it in a sound technical manner.
Technical tools / plugins

- Eclipse \(^{[W7]}\)
  - Spring \(^{[W24][W52]}\)
  - Ant \(^{[W3][W37]}\)
  - JUnit \(^{[W4][W21][W48]}\)
  - Cobertura Coverage \(^{[W6][W40]}\)
  - ICE-faces (JSF-platform) \(^{[W11][W44]}\)
  - SVN \(^{[W39]}\)
  - Apache Tomcat Server \(^{[W4]}\)
- Apache Server \(^{[W3]}\)
- Internet Browsers
  - Mozilla Firefox \(^{[W7]}\)
  - Internet Explorer \(^{[W16]}\)
- TCP-trace \(^{[W26]}\)
- Hudson \(^{[W39]}\)
- SVN \(^{[W39]}\)

Planning

<table>
<thead>
<tr>
<th>Name</th>
<th>Start</th>
<th>End</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internship</td>
<td>4/08/10</td>
<td>7/01/11</td>
<td>40</td>
</tr>
<tr>
<td>Building Company Car System</td>
<td>4/08/10</td>
<td>26/10/10</td>
<td>75</td>
</tr>
<tr>
<td>Setup environment</td>
<td>4/08/10</td>
<td>7/08/10</td>
<td>100</td>
</tr>
<tr>
<td>Declaration of vehicle data</td>
<td>9/08/10</td>
<td>20/08/10</td>
<td>99</td>
</tr>
<tr>
<td>Add relation</td>
<td>20/08/10</td>
<td>7/09/10</td>
<td>99</td>
</tr>
<tr>
<td>Place Order</td>
<td>7/09/10</td>
<td>6/10/10</td>
<td>99</td>
</tr>
<tr>
<td>Admin</td>
<td>6/10/10</td>
<td>15/10/10</td>
<td>100</td>
</tr>
<tr>
<td>Simulate User</td>
<td>6/10/10</td>
<td>8/10/10</td>
<td>100</td>
</tr>
<tr>
<td>Debug application</td>
<td>8/10/10</td>
<td>13/10/10</td>
<td>100</td>
</tr>
<tr>
<td>Managing specials add</td>
<td>13/10/10</td>
<td>15/10/10</td>
<td>100</td>
</tr>
<tr>
<td>Demonstrating and testing</td>
<td>15/10/10</td>
<td>26/10/10</td>
<td>30</td>
</tr>
<tr>
<td>Add relation extension</td>
<td>11/10/10</td>
<td>22/10/10</td>
<td>95</td>
</tr>
<tr>
<td>Show relations</td>
<td>11/10/10</td>
<td>15/10/10</td>
<td>95</td>
</tr>
<tr>
<td>Edit relations</td>
<td>18/10/10</td>
<td>22/10/10</td>
<td>95</td>
</tr>
<tr>
<td>Investigate Java LOG systems</td>
<td>29/09/10</td>
<td>7/01/11</td>
<td>10</td>
</tr>
<tr>
<td>General Project Setup</td>
<td>13/10/10</td>
<td>29/10/10</td>
<td>25</td>
</tr>
<tr>
<td>Handover documentation &amp; presentation</td>
<td>February 2011</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
System analysis

General
Within Nissan Australia, developing will be done in Java in combination with JSF. All code must be tested with JUnit and have a testing coverage of 95% and user approval before it can be taken to production.

The Company Car System will use on top of the default requirements Nissan’s default framework in combination with ICE-faces. The different parts of the system will be developed in a predefined order. The criteria for sorting the parts are difficulty, relations and dependencies, and priority. The developing will be in the following order: Declaration of vehicle data, Add relation (Will be Maintain relation in the new system), Place order, Admin.

System overview
In figure 3 is an overview of how the new system will work, this is not very different from the old system. Only the Session Control Bean, User profile and application configuration are new in the new system. The framework will provide the basic version of the Session Control Bean, user profile and application configuration. There can be an extension written on these basic parts. For the Company Car System there will be custom extensions written for these functions to cover the requested functionality.

Figure 3: A general approach how the system is build up
General module layout

The new system is based on the MVC pattern™ (see figure 4 for an example that implements Java servlets). MVC is at Nissan implemented as follow the jspx page is the view, pageBean the controller and the DAO™, data tables and entities together the model (Figure 5).

![MVC Diagram](image)

Figure 4: Applied MVC pattern on Java servlets

Within the structure of Nissan’s framework the following packages will be used within the new structure of the program. Packages with an asterisk are inherited from the framework.

- au.com.nissan.faces.cc
  - application (Application wide related classes) *
  - dao (Data Access Objects)
  - entities
  - pagebeans (The controllers)
    - ws (Web service classes) *
    - util (General webservice utils)
  - session (Session related classes) *
  - user (User (entity) objects) *

By using this model, each part of the program’s structure is clear for every developer. He or she can easily read and debug the code without excessive research through the code. Also this structure creates an easy entrance for making extensions and modifications to program. If a part of the program is relocated or copied to another project, the only things developers have to do is running the tests and fix project related code.
Figure 5: Basic flow diagram of the class layout

System technologies

Nissans applications are build on a mainframe system with the SOA architecture. The mainframe system communicates via an Enterprise Service Bus (ESB) with SOAP services to the external programs. The SOAP service XML request and response standards are defined per service, based on a standard layout (see below for examples). Each service has a Java library for accessing and using the request and response documents in the web programs.

Table 1: General SOAP request

```xml
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
      «Service specific content»
    </«Servicename»>
  </soapenv:Body>
</soapenv:Envelope>
```

Table 2: General SOAP response

```xml
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
      «Service specific content»
    </«Servicename»>
  </soapenv:Body>
</soapenv:Envelope>
```
Table 3: General SOAP fault

Within the development process of a Nissan application are three phases: development, QA (Quality Assurance) and production. The development phase will contain the project's main build activities. In QA there will be testing, debugging and general maintenance of the application. And in production the end-users can view and use the application. All changes (after the main build) to an application will be done in QA. After testing and approval of the end-users, the application will be deployed to production.

A single project can hold multiple modules, which independently will be developed. The order of development is based on the priority and relation and dependencies. When all modules are completed, the project will be deployed into production.

Figure 6: Nissan development process

Code standards

Within Nissan, there are many code standards. The most important are (in random order):

- Code must be written in a way that it uses as less as possible resources
- Package names all lowercase e.g. "au.com.nissan.<implType>.<systemName>.<useType>.*"
  - implType represents the primary technology we are using with the system e.g. "faces"
  - systemName represents a specific name for the system being implemented
  - useType should reflect the use of the package in the system
- Local variables called (in the code) by their getter
- Code must be tested and have a coverage of 95%
- All environment based properties in an externally accessible configuration folder

These standards will be maintained during the development phase and each web application gets tested by Toxicity Chart[^W11] what’s generating a complexity graph (See attachment III for the Company Car System final graph) based on predefined requirements by Nissan. The web application gets also tested by Hudson[^W10] what will assure that the web application fully is tested and covered before it’s launched to the webservers.
Overview Company Car System

Current system
The current Company Car system replaced paper forms. The paper forms were transferred to HTML forms, whereby the layout and way of processing was similar to each other. The need for changing the paper forms to an electronic environment came from extending the lease types, what generated a lot more requests and a lot more processing time of those requests.

The current system is based on Application Composer, a XML based system that maps the service variables against a pregenerated HTML version of the webpage. The mappings between the webservice and the webapplication is managed by the Altova MapForce\textsuperscript{\cite{W1}} program, which shows how the variables are connected between the two sides. The current system runs on WebSphere\textsuperscript{\cite{W37}}.

The system has two entry points for login. One for the office-staff and one for the plant and warehouse employees. By logging in, the office-staff use their LDAP account and the plant and warehouse people their employee number and date of birth.

In the system is a bar for displaying messages to the user. This bar is located at the bottom part of the system. In this bar the user can read messages about the action he or she has done. These messages can be a general feedback, an error or warning.

New system
The new system will be based on the Nissan web framework in combination with ICE-faces. The SOAP services will be used in combination with their corresponding libraries and entities. All configurable properties will be stored in separate XML files based on Nissan's DataTable model. The system must be configurable for multiple profiles so multiple marques can use it. This means also all direct links to a brand must be setup in a flexible way so it can be linked to the user profile or webservice.

The new system must contain the same functionality as the old system, with some improvements, additions and bug fixes. The improvements include user interface readability, minimize service calls, etc. The additions include debugging, front-end maintenance special-vehicle add, view and update of relations.

Displaying messages in the new system is similar to the old system. All messages that appear in the old system must be in the new system. For each type of message, there are different colours and icons (See table 4: message types).

Table 4: Message types

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>![Info message]</td>
<td>![Warning message]</td>
</tr>
<tr>
<td>![Error message]</td>
<td></td>
</tr>
</tbody>
</table>
Migration of the Company Car System

General
Migration of the Company Car System is not done without thorough thinking through all different scenarios and investigating the old system’s functions. In this chapter, there will be brief descriptions of the different parts of the system. The migrated parts we discuss (See also Figure 7: Menu of Company Cars) are: add relation, place order, declaration of vehicle data (further called FBT Declaration), simulate user and admin divided in 3 sections namely General, Simulate user and Manage Specials. For each main part there will be a general example of one of the possible scenarios. For the admin sections will only be a short description of the functionality. The ways how to enter the program will be discussed at the end of this chapter.

When the system is deployed to the webserver, there will be one webservice called and the response will be saved in the memory. When there are changes to the data of this webservice a Company Car manager can reload it manually via the General section in the admin part of the system.

More detailed screenshots of each part of the system can be found in attachment ‘VII Program screenshots’.

FBT Declaration

In 'FBT declaration’ can the employees for each car they lease declare every month their total driven kilometres for work and private use. Also all expenses they made for the car, divided in petrol and other expenses, will be registered. For every car there will be also some details displayed so the employee can easily identify his or her cars.

In the old system if an employee had no cars (yet) assigned, the system displayed the page without any dynamic data and error message that tells there is no data found. In the new system if an employee had no cars there will be a message that informs the employee instead of displaying a page without data.

The employee has to fill in all required data for every car before he or she can submit it. The required fields in the part of the program are the number of driven kilometres during the given month.

If a car is grounded, the number of kilometres is known and already filled and the field is locked. He or she can only update the rest of the fields to complete the last data for the car.
Scenario - Updating FBT data (Figure 9: FBT process)
A new employee has a Micra as a company car. After the first month he gets an email (Figure 10: FBT request email) with the request to complete his FBT data. He opens the declaration screen via the provided link and sees the details about his vehicle. He fills in the total kilometres of the odometer, the total petrol expenses and other expenses he had for the car and calculates how much he has used the car for private use (Figure 11: Completed FBT screen). After filling in the details he presses save and gets the message back ‘Your FBT Details have been updated’.

Figure 9: FBT process

Figure 10: FBT request email

Figure 11: Completed FBT screen
Add relation

Figure 12: Old add relation screen versus new add relation screen

In ‘Add Relation’, drivers to order a car can be added and modified. The form contains the driver’s personal details, relation with the employee, driver licence details and contact details like an address, a phone number and the company it’s belonging to. The new part will contain the maintain function, so an employee can actually see which relations he / she has and edit them if there is a change in the details. An employee can’t edit the relation type and remove a relation.

By adding a relation the relation must be 18 years or older and have a valid driver licence. For every relation can be 2 addresses filled in and up to 4 different phone numbers. There must be at least one address and phone number provided.
Scenario – Adding a Relation (Figure 13: Add relation process)
A new employee wants to lease a company car. In order to lease the company car he needs to add a relation with the related contact details. The employee goes to maintain relations and clicks on add. After clicking on add the employee selects the company he wants a car for (in this case ‘Nissan’) and select the type of relation (in this case ‘Employee’)(Figure 14: General part of Maintain relations). Then the employee fills all the requested fields in the ‘Relative Information’ screen (Figure 15: Filled Relative information screen). After filling all requested details in, the employee clicks on save and gets the message that the relation is saved successfully.

![Open 'Maintain Relations' Screen](image1)
- Press 'Add' button
- Select company and relation type
- Fill out 'Relative Information'
- Press 'Save' to add the relation

![Figure 13: Add relation process](image2)

![Figure 14: General part of Maintain relations](image3)

![Figure 15: Filled Relative information screen](image4)
Place order

![Old order screen versus new order screen](image)

Figure 16: Old order screen versus new order screen

In this part the employee can see his assigned, allocated and ordered cars. Also he or she can order a car. If the employee has a current car assigned (and has the car collected), he or she can see several details of the car and can (if necessary) order a new one. If a car’s status is allocated means that the car is reserved, but not handed out to the employee.

For ordering a car, the employee can choose between several options. He can choose the car by marque then model series and then the model variant. There is a second way for choosing a car that is by selecting only the special priced cars. In this case, the employee can choose of different model variants of the selected marque. After selecting a variant, the employee can provide 3 colours of his choice, the preferred delivery date, a comment for clarification, accessories he or she wants with the car and the average kilometres per month he or she drives with the car.

An order can be changed until the car is allocated, in this stage the order can’t be changed and also you can’t create a new order for that lease type. After receiving an email with the notification that the car is ready, the employee can collect the car from the company car lot. After collecting the status of the order changes to allocated and assigned. In this stage, you can create a new order to replace the existing car.
If a car is assigned to a lease type and you selected it, the series, colour, registration number and driver will be shown. The select box with all lease types has also in this case the car details and driver for easy identification of which car you want to select.

**Scenario – Place a new order for a car (Figure 17: Place order process)**

A new employee wants to lease a Company Car. After adding a relation, he is ready to add the vehicle. He selects his lease type (Figure 18: Current vehicle details screen) and the order screen gets loaded. The employee has only access to the Nissan marque cars. He selects a Micra Hatch with three possible colours. He fills in all required fields, presses ‘Save’, gets the message back ‘Order added successfully’ and received an email (See VII. Order Confirmation form).

![Figure 17: Place order process](image-url)

![Figure 18: Current vehicle details screen](image-url)

![Figure 19: Filled request order form](image-url)
Admin
In this part of the system, there are several options where the administrator can manage the system. The options are enabling debugging, simulating users, refreshing different static data and managing a side ad to promote specials. The admin part is divided in three parts, namely general, simulate user and Manage Specials

General (Admin Services)
In this part, an administrator of Company Cars can turn debug on and off and dynamic reload services / datatables. When debug is active, a message on the right of the page header will appear. This message tells debug mode is on and the reports will be send to the displayed email address (See figure 12).

Simulate user

![Old simulate screen versus new simulate screen](image1)

Figure 20: Old simulate screen versus new simulate screen

In this part, an administrator of Company Cars can simulate an user selected by username or employee number. The administrator cannot simulate itself. When simulation is active, this page is locked and on the right of the page header is a message that’s telling you that you simulating and who it is (See figure 9&10). An administrator or employee that’s has simulation rights can simulate a user.

![On / Off simulation screen](image2)

Figure 21: On / Off simulation screen

![Header with debug and simulation on](image3)

Figure 22: Header with debug and simulation on
**Manage specials**

In this part, the administrator can for each marque edit the ad (See figure 9) to promote their specials. If there is no content defined the ad is not displayed. The administrator can create also the ad in Word and past it in the input box. The program will strip all unnecessary HTML tags from the Word text.

The ads will be saved in separate files for each marque. During loading the ads in the program setup and a marque-file don’t exist, it will be created as an empty file.

The wysiwyg editor used by ICE-faces is FCKEditor API[^w5]. In 'Figure 24: Wysiwyg editor for editing the ads' you can see an example how the administrator can manage the content of the ads.

---

[^w5]: Fullscreen Content-Komposition Editor

---

**Secondary login**

For all employees that have no LDAP login credentials (mostly employees working at the plant and warehouse) is a secondary login available. By providing the combination of their employee number and date of birth, they can login. After login, they can use the program the same as someone who’s logged in with a LDAP account.

If an employee with a valid LDAP account tries to login, a message appear that they had to use the LDAP login variant.
Quality control

General
Quality control is an important part of development within Nissan. It’s a guarantee that the application is high quality build and reliable. It is also a proof that the end-user is satisfied that he or she can use it without any restraint.

To make this quality control easy to use and clear for everyone, there are three steps that describe this process. These are together called QA and come after the development phase. The steps are ‘Unit Testing’, ‘Acceptance Testing’ and ‘User Testing’ (Figure 15: QA process enlarged).

As mentioned earlier a project can be divided in multiple modules, which have each its own state. Which means there can be multiple QA processes where each module will be individual tested and approved.

Unit testing
Unit testing the application will assure the quality of the code. The combination of unit testing and coverage measuring creates a whole picture of the project’s quality and reliability. As mentioned earlier every class tests must cover 95% of the class’ code. If some parts of the code cannot be tested, the developer must substantiate why the code can’t be tested.

The developer will write the tests during the development of the code. This means each function has a test after finishing a part of code. The development time will be increasing by writing the test immediately, but the other side is that the created code is almost ‘bug free’ and all possible exceptions are caught (See table 5 & 6 for an example).
public boolean checkDOB(Object obj) {
    boolean dob = false;
    if (obj != null && !StringUtils.isBlank(obj.toString())) {
        String DATE_FORMAT = "E MMM dd kk:mm:ss z yyyy";
        String minAge = getSessionFactoryBean().getAppConfigLoader().
                        getDataTableValue("relationData", "dobVals", "minimum");
        try {
            Calendar cal = Calendar.getInstance(getTimeZone());
            if (minAge != null) {
                cal.add(Calendar.YEAR, -Integer.parseInt(minAge));
                cal.add(Calendar.HOUR, 48);
            } else {
                LOG.debug(" >> init >> No minimum age set");
            }
            Calendar cal2 = Calendar.getInstance(getTimeZone());
            cal2.setTime(new SimpleDateFormat(DATE_FORMAT).parse(obj.toString()));
            dob = cal.after(cal2);
        } catch (ParseException e) {
            LOG.error(" >> checkDOB >> ");
            LOG.error(e);
            return false;
        }
        if (!dob) {
            getSessionFactoryBean().getInfoDisplay().queueErrorMessage("Date of birth must be at least " + minAge + " years ago");
            return false;
        }
        return true;
    } else {
        LOG.info(" >> DOB not set");
        return false;
    }
}

Table 5: Example function

@Test
public void testCheckDOB() {
    assertTrue(addRelation.checkDOB("Mon Sep 13 10:00:00 EST 1980"));
}

@Test
public void testCheckDOBTooYoung() {
    assertFalse(addRelation.checkDOB("Mon Sep 13 10:00:00 EST 2000"));
}

@Test
public void testCheckDOBNull() {
    assertFalse(addRelation.checkDOB(null));
}

@Test
public void testCheckDOBInvalid() {
    assertFalse(addRelation.checkDOB("Sep Mon 13 10:00:00 2000 EST"));
}

Table 6: Tests written for example function
The general getters and setters for the different variables will be tested by a separate class, which is added to the test classes that need tests for getters and setters. This test class (GetterSetterTest.java) is written by Steven Grimm. This test class will make it for the developer a lot easier and faster to prove his getter/setter code is covered instead of writing for every getter and setter a proper test (See table 7 to 9 for a getter / setter example).

```java
@Test
class GetterSetterTest {
    UserDetails userDetails = new UserDetails();
    userDetails.setProcessingMonth("2010-06-14");
    assertEquals("2010-06-14", userDetails.getProcessingMonth());
    }
```

Table 7: Private variables UserDetails entity

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>countryID</td>
<td>String</td>
</tr>
<tr>
<td>email</td>
<td>String</td>
</tr>
<tr>
<td>lastUpdate</td>
<td>String</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
</tr>
<tr>
<td>number</td>
<td>String</td>
</tr>
<tr>
<td>processingMonth</td>
<td>String</td>
</tr>
<tr>
<td>processingMonthDisplay</td>
<td>String</td>
</tr>
<tr>
<td>returnDate</td>
<td>String</td>
</tr>
<tr>
<td>returnDateDisplay</td>
<td>String</td>
</tr>
<tr>
<td>vehicles</td>
<td>List&lt;Vehicle&gt;</td>
</tr>
</tbody>
</table>

Table 8: Getter / Setter test source

```java
@Test
class GetterSetterTest {
    UserDetails userDetails = new UserDetails();
    userDetails.setProcessingMonth("2010-06-14");
    assertEquals("June", userDetails.getProcessingMonthDisplay());
    }
```

Table 9: Implementation of excluded variables
Coverage during testing will be done with Cobertura\cite{W6}, so the developer can see during development of all code is covered. Cobertura is used as a plugin of Eclipse (Figure 16: Cobertura Coverage example). By deploying the project to production it will not tested by Cobertura, but tested by Hudson\cite{W10}, which also monitors how much a project deploy fails and how long the projects is running since the last deploy.

![Coverage Session View](image)

**Figure 27: Cobertura Coverage example**

### Acceptance testing

Another developer will do acceptance testing of a module. This testing will proof that all required functionality is implemented. If not the module goes back to the development phase, where it follows the whole process again.

### User testing

User (end) testing will guarantee that the application meets the given requirements. This means that the end-user will check that all required functionality is implemented properly and the application is working as specified.

If a problem is found, the user will write a feedback document that describes the problem including which module and part of the module the problem is found. If possible, a screenshot with directions is added for clarification.

If no problems are found the user writes in the feedback document that the tested module has no problems.
The feedback document can be from a simple note to a full detailed document and everything in between them. The size of the document depends on the complexity and number of problems.

The feedback will be examined to determine if the module is ready or needs to go back to the development phase to repair problems or missing requirements. When the feedback tells the tested module is approved it will be parked until all modules are approved for deployment in production.

If there are problems in production, the application goes back to development and will go through the whole process again. In some urgent cases, the test and approval steps will be passed and the repaired version will be directly deployed to production. Afterwards the fix must be tested and approved by the users to ensure the quality of the program.

Figure 28: Feedback item example
Project Metrics
To ensure the code is high quality, not only for the user and for the unit tests, a metric tool will measure it. This process forces the developer to write his / her code as optimal as possible.

This means the developer keep track of all of his code and generic functions will be placed in a separate class to provide all other classes’ access to it and make it clear to other developers.

I’ve measured the Company Car project with the following aspects, lines of code, comments lines of code and complexity. The measurements are grouped per part to make the overview readable. Only the three main parts will be discussed.

I’ve used SourceMonitor to measure the metrics. In Figure 29, you can see a Kiviat Metrics Graph of the system created by SourceMonitor. The green ring in the graph is the optimal value that can be achieved for each metric. The result in this graph meets my expectations. I will explain max complexity, average statements per method and block depth. Max complexity is that high because the business logic makes the order part very hard to implement in simple code statements. There are some small functions that are used generic through the whole application. These functions make other code easier to read. This makes the average statements very low. The block depth is a little bit high, because the most complex part of the program has many decisions in it to comply with the business rules.

![Kiviat Metrics Graph of Company Car System](image-url)

Figure 29: Kiviat Metrics Graph of Company Car System
**Code complexity**

In figure 30, you can see the average complexity of each part of the program. This result confirms my expectation that the order vehicle is the most complex part of the program. It surprised me that the 'FBT declaration' is almost equal to the 'Maintain relations' part. I expected here that 'Maintain relations' is more complex because of all the business rules that apply to that part. The only reason I can think of why this is almost equal is that a lot of common functions in Maintain relations are grouped together and simplified.

![Complexity graph](image)

Figure 30: Complexity graph

**Comment** Lines of Code

In figure 31, you can see the number of lines per part. The average lines of code per part are in the range I expected to be. I used the comments as a brief description of functions behaviour. By making the descriptions brief, the comments are within the margins used in Software engineering.

![Lines of Code graph](image)

Figure 31: (Comment) Lines of Code

**Complexity graph**

In attachment 'III. Company Car System complexity graph', you can see a graph with the classes that have the biggest values for the given metrics. This graph shows again that the 'Order' part is the most complex part followed by the 'Maintain Relation' part.
Conclusion
The whole process of going abroad and doing an internship was a great experience. Spending a long time (approximately 6 months) in a foreign country, gives you the opportunity to learn the local language and culture. I can say you must do it to enriching your life.

Working in Australia is different than here in the Netherlands. The work is less hectic and you get encouraged to take risks for achieving the best results. The standards within the company where easy to adapt, because it joined to everything I learned during my study in Software Engineering.

The methods used during the project were a revelation for me. It was an opportunity to see the learned theory work in real. Within the main project, I used MoSCoW to define what the main functionality was and what was optional. In the next project with scrum, (See attachment 'V. Other tasks') I worked more with MoSCoW. This project showed me a way to implement daily stand-up meetings with the business. It also showed me how to prioritize and estimate time for a task.

I took the opportunity, while I was in Australia, to see a part of it. During the travel, I tried to understand the way of life in Australia. The country is so big that every state has its own culture and history. The residents of Australia travel easier long distances with a car, plane and public transport. I've managed to adapt a little bit to this standard so I could see the most of the country in a short time period.

In the period I was there, there were a number of major natural disasters. The natural disasters were more intensive than normal, but the Aussies acted almost as if it was normal. This behaviour comes from the fact that every year there are a few of these natural disasters in those areas. The local government has taken a number of actions to limit the damage in the area. A few of those actions are warning people how big the change is of a disaster (See Figure 32) and make everybody aware of the danger around there living area.

Figure 32: Fire rating sign, an example how the local government warns the people

I've learned, from this internship and travelling around, to react with a dose of common sense and humour to daily problems. Also I learned apply everything I learned during my study into a big company with existing standards.
Literature list

Web

[W1] Altova MapForce – Graphical Data Mapping, Conversion, and Integration Tool
http://www.altova.com/mapforce.html

[W2] Apache – Ant
http://ant.apache.org/

[W3] The Apache Software Foundation
http://www.apache.org/

[W4] Apache Tomcat
http://tomcat.apache.org/

[W5] CKEditor - WYSIWYG Text and HTML Editor for the Web
http://ckeditor.com/

[W6] Cobertura – Sourceforge
http://cobertura.sourceforge.net/

[W7] Eclipse - The Eclipse Foundation open source community website
http://www.eclipse.org/

[W8] eNode - Powerful XML Infrastructure
http://www.eNode.com

[W9] Hogeschool van Amsterdam
http://www.hva.nl/

[W10] Hudson - Extensible continuous integration server
http://hudson-ci.org/

[W11] How toxic is your code?

[W12] ICEfaces - Open Source Ajax, J2EE Ajax, JSF Java Framework
http://www.icefaces.org/

Expresso Developer’s Guide – Chapter 4 What is MVC?

[W14] JUnit – Sourceforge
http://junit.sourceforge.net/

http://www.altova.com/mapforce.html

[W16] Microsoft Internet Explorer

[W17] Mozilla project
http://www.mozilla.org/

[W18] Nissan Australia: Shift The Way You Move. Cars, 4x4’s, Dealers
http://www.nissan.com.au

[W19] Nissan Motor Company Global Website
http://www.nissan-global.com

http://www.oracle.com/technetwork/java/javasee/javaserverfaces-139869.html

[W21] Plaintive Mewling - Utility class for simple getter/setter JUnit testing
http://www.plaintivemewling.com/?p=34
[W22] Renault.com - Car manufacturer Renault’s official international website - Renault
http://www.renault.com/en/Pages/home.aspx

[W23] Scrum Methodology
http://scrummethodology.com/scrum-sprint/

[W24] Spring
http://www.springsource.org/

http://java.sun.com/products/jsp/

[W26] TCP-Trace
http://www.tcptrace.org/

[W27] Webopedia, API
http://www.webopedia.com/TERM/A/API.html

[W28] Webopedia, JSP
http://www.webopedia.com/TERM/J/JSP.html

[W29] Webopedia, LDAP
http://www.webopedia.com/TERM/L/LDAP.html

[W30] Webopedia, Model-View-Controller
http://www.webopedia.com/TERM/M/Model_View_Controller.html

[W31] Webopedia, Rich Internet Application
http://www.webopedia.com/TERM/R/Rich_Internet_Application.html

[W32] Webopedia, Service-Oriented Architecture
http://www.webopedia.com/TERM/S/Service_Oriented_Architecture.html

[W33] Webopedia, SOAP

[W34] Webopedia, TCP
http://www.webopedia.com/TERM/T/TCP.html

[W35] Webopedia, URL
http://www.webopedia.com/TERM/U/URL.html

[W36] Webopedia, XML
http://www.webopedia.com/TERM/X/XML.html

[W37] WebSphere software
http://www-01.ibm.com/software/nl/websphere/

[W38] Wikipedia, the free encyclopedia, Apache Ant
http://en.wikipedia.org/wiki/Apache_Ant

[W39] Wikipedia, the free encyclopedia, Apache Subversion
http://en.wikipedia.org/wiki/Apache_Subversion

[W40] Wikipedia, the free encyclopedia, Code coverage

[W41] Wikipedia, the free encyclopedia, Data access object

[W42] Wikipedia, the free encyclopedia, Enterprise service bus
http://en.wikipedia.org/wiki/Enterprise_service_bus

[W43] Wikipedia, the free encyclopedia, Fringe Benefits Tax (Australia)
http://en.wikipedia.org/wiki/Fringe_Benefits_Tax_(Australia)

[W44] Wikipedia, the free encyclopedia, ICEfaces
http://en.wikipedia.org/wiki/ICEfaces
[W45] Wikipedia, the free encyclopedia, JavaServer Faces

[W46] Wikipedia, the free encyclopedia, JavaServer Pages

[W47] Wikipedia, the free encyclopedia, Jspfx

[W48] Wikipedia, the free encyclopedia, JUnit
http://en.wikipedia.org/wiki/JUnit

[W49] Wikipedia, the free encyclopedia, Model–View–Controller
http://en.wikipedia.org/wiki/Model-View-Controller

[W50] Wikipedia, the free encyclopedia, Nissan Motors
http://en.wikipedia.org/wiki/Nissan

[W51] Wikipedia, the free encyclopedia, Scrum (development)

[W52] Wikipedia, the free encyclopedia, Spring Framework

http://www.w3.org/

Books
Core JavaServer™ Faces, Second Edition - Prentice Hall

Pro JSF and Ajax: Building Rich Internet Components – Apress

Java phrasebook, essential code and commands – Developer's library

Java Web Services – O'Reilly
<table>
<thead>
<tr>
<th>Word</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ant</td>
<td>software tool for automating software build processes</td>
</tr>
<tr>
<td>API</td>
<td>Application Program Interface, a set of routines, protocols, and tools for building software applications</td>
</tr>
<tr>
<td>DAO</td>
<td>Data Access Object, an object that provides an abstract interface to some type of database or persistence mechanism, providing some specific operations without exposing details of the database. It provides a mapping from application calls to the persistence layer</td>
</tr>
<tr>
<td>ESB</td>
<td>Enterprise Service Bus, a software architecture construct which provides fundamental services for complex architectures via an event-driven and standards-based messaging engine (the bus). Developers typically implement an ESB using technologies found in a category of middleware infrastructure products, usually based on recognized standards</td>
</tr>
<tr>
<td>FBT</td>
<td>Fringe Benefits Tax, a tax applied within the Australian tax system by the Australian Taxation Office. The tax is applied to employers who provide certain &quot;fringe benefits&quot; to employees</td>
</tr>
<tr>
<td>ICE-faces</td>
<td>ICE-faces is an open source Ajax framework that enables Java EE application developers to create and deploy server-based rich Internet application (RIA) using the Java language. ICE-faces is based on JSF.</td>
</tr>
<tr>
<td>JSP</td>
<td>JavaServer Faces Technology, a standard for building server-side user interfaces</td>
</tr>
<tr>
<td>JSPx</td>
<td>XML version of jsp</td>
</tr>
<tr>
<td>JUnit</td>
<td>JUnit is a simple framework to write repeatable tests. It is an instance of the xUnit architecture for unit testing frameworks</td>
</tr>
<tr>
<td>LDAP</td>
<td>Lightweight Directory Access Protocol, a set of protocols for accessing information directories</td>
</tr>
<tr>
<td>MVC</td>
<td>Model-View-Controller, the term used to describe a software paradigm that separates objects into three categories</td>
</tr>
<tr>
<td>RIA</td>
<td>Rich Internet Applications, Web-based applications that function as traditional desktop applications however Web browsers (or clients) are required for access but unlike traditional applications, software installation is not required</td>
</tr>
<tr>
<td>Scrum</td>
<td>Scrum is an agile software development project management methodology. The method is based on iterative and incremental steps, also known as sprints.</td>
</tr>
<tr>
<td>SOA</td>
<td>Service-Oriented Architecture, an application architecture in which all functions, or services, are defined using a description language and have invokable interfaces that are called to perform business processes</td>
</tr>
<tr>
<td>SOAP</td>
<td>Simple Object Access Protocol, a lightweight XML-based messaging protocol used to encode the information in Web service request and response messages before sending them over a network</td>
</tr>
<tr>
<td>Spring</td>
<td>Spring Framework is an open source application framework for the Java platform</td>
</tr>
<tr>
<td>SVN</td>
<td>Apache Subversion, a software versioning and a revision control system</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol, one of the main protocols in TCP/IP networks</td>
</tr>
<tr>
<td>URL</td>
<td>Uniform Resource Locator, the global address of documents and other resources on the World Wide Web</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language, a specification developed by the W3C</td>
</tr>
</tbody>
</table>
Attachments

I. Application letter internship .......................................................... 36
II. Application for graduation assignment (Dutch) .................................. 37
III. Company Car System complexity graph .......................................... 38
IV. General planning ........................................................................... 39
V. Other tasks ...................................................................................... 42
VI. Program screenshots ....................................................................... 43
VII. Order Confirmation form ................................................................. 51
I. **Application letter internship**

All applications requests are based on the basic letter below. Each request has a little addition at the first paragraph for the company why I wrote to them.

---

Arnoud Sanderse  
Melil 50  
1985 ES HEEMSKERK  
Noord-Holland, Netherlands  
Skype: ArnoudSanderse  
E-mail: info@arnoudsanderse.nl  
Website: [http://www.arnoudsanderse.nl](http://www.arnoudsanderse.nl)

Dear Sir / Madam,

Let introduce myself, I am Arnoud Sanderse, male, 23 years old and live in Heemskerk, the Netherlands.

Since my childhood I have had a dream about going to Australia. For several years I have wanted to go to Australia and with this opportunity I can realise my dream in combination with getting abroad working experience. A nearshoring project on my school showed me how important it is to get experience with working with other cultures.

Currently I am studying computer science at the Hogeschool van Amsterdam and will earn a Bachelor Degree in ICT. I specialize in Software Engineering. During my studies I have developed several skills like programming in different languages, computer networking, project management methods, software testing, requirement engineering, interface design, data warehousing and industrial automation.

My work experience in the IT business involves making websites for voluntary organisations and repairing computers for friends. I have done two internships, one in a hospital maintaining the IT networks and one in a printing office doing helpdesk work and making a custom made publishing (web)application. I have done several IT projects at school which were mostly projects to build (web)programs for a client.

The field I want to do my internship in is security solutions during development and test-driven development. I hope to learn how to setup a bug free application.

I would like to do an internship of 100 working days between July and December 2010.

With kind regards,  
Arnoud Sanderse

Attachments:  
- Curriculum Vitae (resume)
II. Application for graduation assignment (Dutch)

---

### Aanmeldingsformulier Afstudeeropdracht

#### Persoonlijke gegevens
- Voornaam tussenvoegsel & student: Arnoud Sanderse
- Studentnummer: 535766
- Opleiding: Voort Z Deel tijd  T Meesterwerk
- Adres student: Merel 50, 1965ES Heemskerk
- Telefoonnummer: +31 6 4244 7578
- Naam beoogde examinator: Richard de Koning
- Naam tweede lezer: 
- Naam bedrijf: NISSAN MOTOR CO. (AUSTRALIA) PTY LTD.
- Postadres bedrijf: 260 - 284 Frankston Dandenong Road
- Postcode + Plaats: Dandenong South, VIC, 3175, Australia
- Bezoekadres bedrijf: 260 - 284 Frankston Dandenong Road
- Postcode + Plaats: Dandenong South, VIC, 3175, Australia
- Telefoon / email / website: +61 3 9797 4169 Jay_brunett@nissan.com.au  
  http://www.nissan.com.au
- Naam bedrijfsbegeleider: Jay Brunett
- Emailadres bedrijfsbegeleider: Jay_Brunett@nissan.com.au
- Aanvangsdatum: 4 augustus 2010
- Beoogde afstudieraad: 1 januari 2011

### Verantwoording goedkeuring afstudeeropdracht

- Externe orientatie
- Oriënterende interviews
- Analyse

### Omschrijving afstudeeropdracht

#### Titel
- Nissan Company Car Order system

#### Omschrijving van de opdracht
- First stage of the assignment is to redevelop and enhance the current version of the Nissan Company Car Order System into an ajax enabled JSF Web application. After this, the second stage will be to help with the development of other Nissan core web based applications.

#### Belangrijkste leerdoelen
- To understand the business requirements for the company car system, and deliver it in a sound technical manner.

#### Wat wordt er w.b. technische competenties geleerd, c.q. aangevoerd
- Java / JSF
- Use and integration of Web Services
- XML Technologies
- Reading and Interpreting other team members code
- Debugging problems

#### Wat wordt er w.b. niet-technische competenties geleerd, c.q. aangevoerd
- Working in a collaborative Team environment
- Documentation
- Use and abiding to Standards
- Presenting work to team members in weekly project meetings

### Business unit
- SE, SNE, ITM, TC, HCD, overg

---

1 Ter goedkeuring door de afstudeercommissie formulier in doc formaat per e-mail indienen via afstudeercommissie-informatica@hva.nl. De afstudeercommissie vergadert ca 1x per 2 weken, zie voor vergaderdagen het jaarrooster https://intra.informatica.hva.nl/intra/rooster. Uiterlijk 2 werkdagen voor de vergadering inschrijven. Na afleveren van de uiteindelijke versie van het formulier voorzien van de benodigde handtekeningen leveren bij de praktijkbureau, BT.26.
2 Examinator is verantwoordelijker voor tweede lezer (alle invullen indien bekend)
3 Zie voor afstudeerzittingenperiode: het jaarrooster op het intranet of de studiegids
4 Meer info: handleiding afstudeeren, https://intra.informatica.hva.nl/content/links/formulieren/
5 BE, SE, SNE, ITSM, GT, ES, IC, FIS

---

Aanmeldingsformulier Afstudeeropdracht
III. Company Car System complexity graph
IV. General planning

<table>
<thead>
<tr>
<th>Internship</th>
<th>Building Company Car system</th>
<th>Setup environment</th>
<th>Declaration of vehicle data</th>
<th>Add relation</th>
<th>Place Order</th>
<th>Admin</th>
<th>Simulate User</th>
<th>Debug application</th>
<th>Managing specials add</th>
<th>Demonstrating and testing</th>
<th>Add relation extension</th>
<th>Show relations</th>
<th>Edit relations</th>
<th>Investigate Java LOG systems</th>
<th>General Project Setup</th>
</tr>
</thead>
<tbody>
<tr>
<td>Week 31</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Week 32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Week 33</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Week 34</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Week 35</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Week 36</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Week 45</td>
<td>Week 46</td>
<td>Week 47</td>
<td>Week 48</td>
<td>Week 49</td>
<td>Week 50</td>
<td>Week 51</td>
<td>Week 52</td>
<td>Week 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>-------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**December 2010**

**January 2011**

100%
V. Other tasks

Next to the main project, I was supposed to join other tasks within the company. The first tasks I got were improving the basic setup for every project and develop some parts of the new system as generic as possible so it could be reused in other projects.

A little example of a generic part is a page where every configuration or preloaded data can be reloaded (See also Figure 33). This part is situated in the admin section. It creates for the administrator a way of reloading data without redeploying the whole application.

![Figure 33: Reload services screen](image)

When I finished all parts of the project (start of November), I had to wait for the user tests to be done. The tests were scheduled for the end of December. During this period (and after it) I had the opportunity to join a new project with scrum as project method. I joined this project and worked as a developer until my internship ended.
VI. Program screenshots

Login screens

LDAP

![LDAP screenshot](image)

Webservice old

![Webservice old screenshot](image)

Webservice new

![Webservice new screenshot](image)
Home

Old program

New program
Declaration of vehicle data

**Old program**

![Old program screenshot](image1)

**No records found**

![No records found screenshot](image2)

**Declaration form example**

![Declaration form example screenshot](image3)

**Add relation**

**Old program**

![Add relation screenshot](image4)

**New program**

![New program screenshot](image5)
Place order

Old program - New order
Old program - Modify order

Select vehicle lease type
Create new order

Modify order
Remove order

Admin
Old program

General

Simulate User
Manage Specials

There are limited numbers available.
All vehicles must be delivered to the user by 30th November 2011.
Current company vehicles over 5,000Kms can be exchanged over in order to obtain a special, as detailed above.
IB Students will be entitled to special prices.
Contractors with terms of 6 months and over will be entitled to special prices.

Not authorised page
VII. Order Confirmation form

<table>
<thead>
<tr>
<th>SUO Number</th>
<th>0000000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Employee Number</td>
<td>000000</td>
</tr>
<tr>
<td>Employee Name</td>
<td>ARNOUD SANDERSE</td>
</tr>
<tr>
<td>Department</td>
<td>8001 NMA</td>
</tr>
<tr>
<td>Driver</td>
<td>ARNOUD SANDERSE</td>
</tr>
<tr>
<td>Current Vehicle Registration</td>
<td>NONE</td>
</tr>
<tr>
<td>Order Type</td>
<td>L / CONTRACTOR LEASE</td>
</tr>
<tr>
<td>Model Description</td>
<td>MICRA HATCH 1.4 AUTO 4-F</td>
</tr>
<tr>
<td>Colour</td>
<td>TOKYO SILVER</td>
</tr>
<tr>
<td>Model Code</td>
<td>K12</td>
</tr>
<tr>
<td>Standard Accessories Req'd.</td>
<td></td>
</tr>
<tr>
<td>Date Replacement Vehicle Req'd.</td>
<td></td>
</tr>
<tr>
<td>Estimated Lease Charge</td>
<td>$800 per month</td>
</tr>
</tbody>
</table>

Note: Your Current Vehicle should not exceed 12 Months or 10,000km

Your selection of vehicle for assigned/lease/family lease cannot be guaranteed as supply depends on the availability of stock and dealer allocation needs.

This form is a confirmation of your order and will be processed accordingly, you will receive further information upon vehicle assignment.

The lease charge quoted is estimated based on current retail prices and may be subject to change. Exception to this is if a special price vehicle has been selected at the time of order.